Advant® OCS
with Master software

PC Elements
Advant® Controller 160

Reference Manual

3BDS 005 557R101

Use of **DANGER**, **WARNING**, **CAUTION**, and **NOTE**

This publication includes, **DANGER**, **WARNING**, **CAUTION**, and **NOTE** information where appropriate to point out safety related or other important information.

DANGER	Hazards which could result in severe personal injury or death
WARNING	Hazards which could result in personal injury
CAUTION	Hazards which could result in equipment or property damage
NOTE	Alerts user to pertinent facts and conditions

Although **DANGER** and **WARNING** hazards are related to personal injury, and **CAUTION** hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all **DANGER**, **WARNING**, and **CAUTION** notices.

TRADEMARKS

ABB Master is a registered trademark of ABB Asea Brown Boveri Ltd. Switzerland.

Advant is a registered trademark of ABB Asea Brown Boveri Ltd. Switzerland.

MODBUS is a registered trademark of Gould Electronics Inc.

Siemens and all Siemens-products mentioned in this publication are trademarks of Siemens AG.

NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Automation Products GmbH. ABB Automation Products GmbH assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Automation Products GmbH be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Automation Products GmbH be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Automation Products GmbH’s written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose.

The software described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Copyright © ABB Automation Products GmbH 2000

TABLE OF CONTENTS

Introduction

PC Language 1
Call 1
Call Name.....1
Call Parameters 2
Graphic Symbol 2
Connection of PC Element Inputs and Outputs 2
Error Handling, Integer Overflow.....3
Overflow Real Numbers 3
Element Libraries for MasterPiece Products.....3
Programming with the PC Language 3
The Data Sheets.....4
PC Elements According to Type 5
PC Element Short Description 10
Product Grouping 14
Execution Times.....15
Space Requirements 19
Space Requirements for Inline Coded PC Elements.....24
Generation of Inline Code 24

Data Sheets

ABS 25
ADD 26
ADD-MR.....27
AND, OR.....29
AND-O 31
AND-TRG.....33
BGET 34
BLOCK 35
BSET 36
COMP-B 37
COMP-I.....38
COMP-R 39
CON-PU1.....41
CONTRM.....48
CONV 51
CONV-AX.....52
CONV-BI 54

CONTENTS (continued)

CONV-IB 58
COUNT 61
DER 63
DIV 66
DIV-MR 67
FIFO 68
FILT-1P 70
FUNCM 73
FUNG-1V 74
GRC 76
INT 87
INV 90
LIM-N 91
MASTER 94
MAX 97
MIN 99
MONO 101
MONO_UT 103
MOON 105
MOVE 107
MOVERED 108
MUL 109
MULDIV 110
MUX-I 111
MUX-MI 113
MUX-MN 115
OR-A 117
OR-TRG 120
OSC-B 121
PB-IN 122
PB-OUT 123
PC_AX646 124
PC_DX610 128
PDP 132
P-1 135
PI 137
PID 141
PIP 145

CONTENTS (continued)

P_STEP 148
RED_STAT 152
REG 153
REG-G 155
RSM 158
SEQ 160
SHIFT 165
SLAVEM 167
SQRT 168
SR 169
SR-AA, SR-AO, SR-OO, SR-OA 170
SR-D 171
STEP 172
SUB 178
SW 179
SW-C 180
SW-RED 181
SYSL 182
SYSTIME 183
SYS-CRC 184
THRESH-L 186
TIMER 187
TRAC 191
TON, TOFF 193
TON-RET 195
TRIGG 197
XOR 198

Introduction

Advant Controller 160 is a range of products for control of various power plant processes. The system provides the ABB Master Programming Language (AMPL) methods. AMPL is based on software implemented function blocks, called PC Elements, which are connected to each other into programs which form a complete control function. AMPL is used in all Advant Controllers, in MasterPiece 200/1, and MasterPiece 90.

PC Language

The PC element is the smallest unit of the AMPL. Each element performs a complete function, such as a logical AND gate, a timer, or a feedback control element.

Call

Each element has its own call name that enables it to be accessed regardless of the computer system the element is in. Many of the elements have call parameters which allow size, function, data type, and other properties to be determined.

Call Name

The mnemonic call name provides information about the basic function of the element. Further information about the function of the element or the data types for which the element is intended is given by a suffix to the name. For example COMP-I, comparator for integers, and COMP-R, comparator for real numbers, see the table below.

Suffix	Description
B	Boolean
I	Integer
R	Real number
T	Time
G	Group data
M	Memory function Multiplier in compound elements
L	Load function, Limiting function, Threshold function
RW	Read/Write
N	1-of-N addressing
D	Data
P	Number of poles
V	Variable

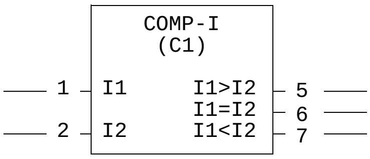
Suffix	Description
O	OR function in compound elements
A	AND function in compound elements

Call Parameters

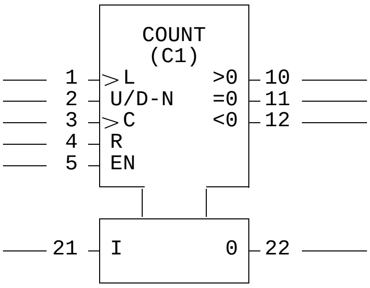
For structure elements the call parameters determine cyclicity, place in the cycle time table (determines the order of execution for PC modules having the same cyclicity) and, for a slave element, the identity of its master header. For a function element, the call parameters may specify the size, data type, number of inputs and outputs, and so on.

Graphic Symbol

Each PC element has a graphic symbol similar to the one in the figure below.



Normally, the call name of the element is written in the uppermost part of the symbol.
For certain functions, for example AND and OR, the name is replaced by a symbol. When the element has supervisory control signals, the terminals for these signals are collected in a separate part of the symbol according to the figure below:



Connection of PC Element Inputs and Outputs

Only inputs and outputs with the same data type can be interconnected. If, for example, an output with data of the type I is to be connected to an input of the type R, a code conversion element, for example CONV (I, R) must be inserted in the data path.

Designation	Data Type
I	Integer, 16 bit
IL	Integer, 32 bit
R	Real number

Designation	Data Type
B	Boolean
T	Time
TR	Time, real representation
G	Group data, that is a collection of data with one of the above data types
A	Array

Error Handling, Integer Overflow

The range of allowable integer values is as follows.

For I = -32,768…+32,767

For IL = -2,147,483,648…+2,147,483,647

When the range is exceeded the output is set to the respective limiting value.

Overflow Real Numbers

The range of allowable real numbers is 9.2×10^{18}…-0.1×10^{18}. When the range is exceeded the output is set to the respective limiting value. The smallest magnitude that can be represented by a real number is 5.0×10^{-20}. Any output with an absolute value smaller than this is set to 0.0.

Element Libraries for MasterPiece Products

Each standard system has an element library which contains a defined subset of the total number of available PC elements. The Technical Description contains the definition of the PC library.

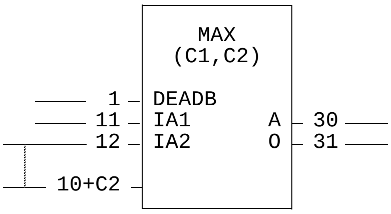
Programming with the PC Language

When programming a PC based control system, one chooses PC elements from the library, and includes them in a draft PC diagram. The elements are then interconnected on the diagram to define the exchange of data between them.

The information on the draft PC program is used when entering the PC program in source code form on a system aid. The aid translates the program into object code, which is loaded into the target system.

The Data Sheets

An element description starts with a short **Summary** that describes the main function of the element. The graphic symbol of the element is shown to the right of the summary. Symbols with a variable number of connections are shown in the figure below:



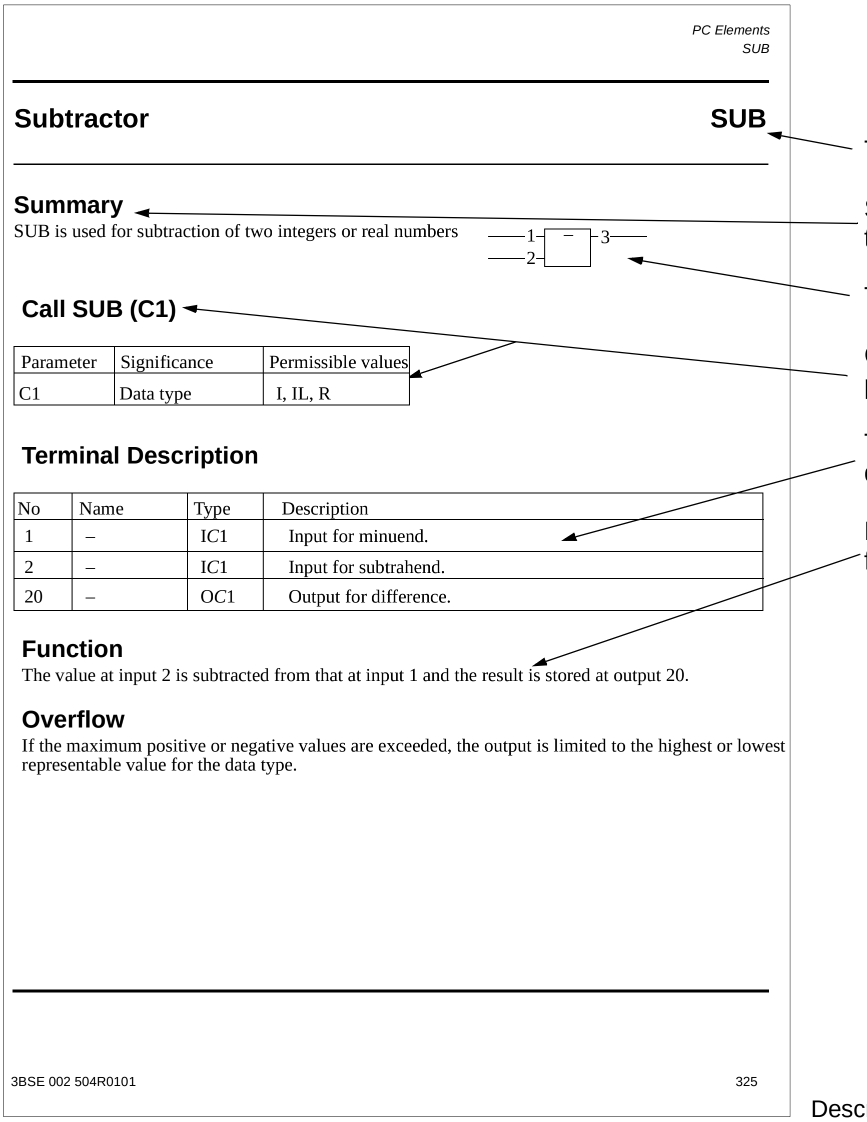
The calling of the element is described under the heading **Call.**

ELEMENT (C1, C2, …, Cn)

If the element has call parameters, these are described in a table where their permissible values are stated. Element terminals are described under the heading **Connections**. This section provides a table giving each terminal number, name, type and application.

The type of connection is given as xy, where x is I or O if it is an input or an output terminal, Y denotes the data type previously described in the table below the heading Connections of PC Elements. The descriptions finish with a detailed functional description under the heading **Function**.

.



Subtractor **SUB**

Summary

SUB is used for subtraction of two integers or real numbers

Call SUB (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	–	IC1	Input for minuend.
2	–	IC1	Input for subtrahend.
20	–	OC1	Output for difference.

Function

The value at input 2 is subtracted from that at input 1 and the result is stored at output 20.

Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

The call name of the element.

Short description of the function of the element.

The graphic symbol of the element.

Call, with table over the call parameters, if any.

The terminals of the element are described in a table.

Detailed description of the elements function.

Description, schematic outline

PC Elements According to Type

Element Type	Element Name
Structure Elements	
Block Header	BLOCK
Control Module Header	CONTRM

PC Elements According to Type (Continued)

Element Type	Element Name
Function Header	FUNCM
Group Control	GRC
Master Header	MASTER
Program Header	PCPGM
STEP Function	P_STEP
Sequence Header	SEQ
Slave Header	SLAVEM
Step Header	STEP
Logic Elements	
And Gate	AND
Inverter	INV
Or Gate	OR
Threshold Function	THRESH-L
Exclusive Or Gate	XOR
AND with OR gates at the inputs	AND-O
OR with AND gates at the inputs	OR-A
AND with trigger on output	AND-TRG
OR with trigger on output	OR-TRG
Bitwise Operations on Integer Data Types	
Set one bit	BSET
Read one bit	BGET
Memory Elements	
Set/ Reset Memory	SR
Set/ Reset Memory with Data Input	SR-D
Set/ Reset Memory with And Inputs	SR-AA
Set/ Reset Memory with And/Or Inputs	SR-AO
Set/ Reset Memory with Or/And Inputs	SR-OA
Set/ Reset Memory with Or Inputs	SR-OO

PC Elements According to Type (Continued)

Element Type	Element Name
Time Elements	
Monostable	MONO
Monostable Untracked	MONO_UT
Oscillator	OSC-B
Time Selector	TIMER
Time Delay Off	TOFF
Time Delay On	TON
Time Delay On with Retention	TON-RET
Trigger	TRIGG
Register Elements	
Queue Register	FIFO
Data Copying	MOVE
Register	REG
Group Data Register	REG-G
Shift Register	SHIFT
Switch Elements	
Switch	SW
Changeover Switch	SW-C
Multiplexers	
Multiplexer with Integer Address	MUX-I
Multiplexer with Memory and Integer Address	MUX-MI
Multiplexer with Memory and 1 of N Address	MUX-MN
Converter Elements	
Code Converter	CONV
Code Converter, Boolean to Integer	CONV-BI
Code Converter, Integer to Boolean	CONV-IB
Arithmetic Elements	
Absolute Value	ABS

PC Elements According to Type (Continued)

Element Type	**Element Name**
Adder	ADD
Divider	DIV
Limiter	LIM-N
Multiplier	MUL
Square Root	SQRT
Subtractor	SUB
Adder with Multiplied Output	ADD-MR
Divider with Multiplied Inputs	DIV-MR
Integer Scaling Element	MULDIV
Counter Elements	
Counter	COUNT
Comparator Elements	
Comparator for Boolean Values	COMP-B
Comparator for Integer	COMP-I
Comparator for Real Numbers	COMP-R
Maximum Selector	MAX
Minimum Selector	MIN
Function Generators	
Function Generator with 1 Variable	FUNG-1V
Feedback and Filter Elements	
Three State Controller	CON-PU1
Derivator	DER
PID-Algorithm	PID
Regulatory Control	
Filter with 1 Pole	FILT-1P
Integrator	INT
P Function Proportional	P-1
PDP Function	PDP

PC Elements According to Type (Continued)

Element Type	Element Name
PI-Regulator	PI
PIP-Function	PIP
Ramp Generator	RAMP
PROFIBUS Communication Elements	
Input from PROFIBUS	PB-IN
Output to PROFIBUS	PB-OUT
Supervision Elements	
Calculation of System Load	SYSL
Element for Actual Time and Date	SYSTIME
Redundancy Elements	
Move for Redundancy	MOVERED
Switch for Redundancy	SW-RED
Redundancy Status	RED_STAT
Tracking	TRAC
Signal Selector	
M out of N Function	MOON
Measurement	
Rotational Speed Measurement	RSM
Protection and Safety Applications	
Manage Fingerprints	SYS-CRC
Converts AX646 Analog Signals to Binary	CONV-AX
Test AX646 Module	PC_AX646
Test DX610 Module	PC_DX610

PC Element Short Description

Element	Description
ABS	Calculates the absolute value of an integer or a real number. The result is multiplied by a scale factor.
ADD	Adds up to 19 integers or real numbers.
ADD-MR	Adds two groups of up to 19 real numbers each and subtracts the second from the first. The result is multiplied by a real number.
AND	Gives the Boolean product of up to 19 Boolean values.
AND-O	Gives the Boolean product of up to 9 Boolean values and up to 5 Boolean sums of a maximum of 9 Boolean values each.
AND-TRG	Used when the output of an AND gate should be triggered.
BGET	Reads the value of the selected bit from an integer or integer long value
BLOCK	Module header for blocks.
BSET	Sets the value of the selected bit in an integer or integer long value
COMP-B	Used to compare two bits.
COMP-I	Comparator for Integer.
COMP-R	Used to compare two real numbers.
CON-PU1	Three state controller with outputs for increase/decrease for control via an integrating actuator.
CONTRM	Module header for control modules.
CONV	Conversion of data between the types I, IL, R, T and TR.
CONV-AX	Convert AX646 Analog Signals to Binary.
CONV-BI	Converts data in BC, BCD, 1-of-N code, Gray code, and 16/32 bit packing, given at a number of Boolean inputs into data in integer form (I or IL).
CONV-IB	Converts data in integer form (I or IL) into data in BC, BCD, 1-of-N code and uncoded unpacking at a number of Boolean inputs.
COUNT	Pre-settable counter for counting pulses, up or down. With checking of the value relative to 0.
DER	Used to give derivation effect. The element has inputs and outputs for limit values, setting of gain, filter function for restricting the derivation effect and the possibility of following and external reference.
DIV	Used for dividing two real numbers or two integers. For integers the quotient is obtained along with a remainder at a special output.
DIV-MR	Used to divide one product of real numbers with another.

PC Element Short Description (Continued)

Element	Description
FIFO	Queue register with up to 9 queues each with a maximum of 64 places.
FILT-1P	1-pole low-pass filter with options for limiting the output signal and for following an external reference.
FUNCM	Module header for function module.
FUNG-1V	Generates a piece-wise linear function of one variable using tables, with up to 255 elements. If the function is strictly monotonous the inverse of the function can be generated by balancing.
GRC	Central control of a group control. Used as a sequential control or a logical control. It has to be used as the header of a step program with P_STEP.
INT	Used to give integration effect. The element has inputs and outputs for limit values, setting of gain and following with an external reference.
INV	Used to invert a Boolean value.
LIM-N	Used for limiting integers, real numbers or time values. Several limit values may be selected.
MASTER	Module header for master modules.
MAX	Selects the largest of up to 19 integers or real numbers.
MIN	Selects the smallest of up to 19 integers or real numbers.
MONO	Used to generate a pulse with a given duration when the input changes from 0 to 1. The pulse duration is specified at an input. Retriggability is controlled with an input.
MONO_UT	Untracked Monostable. For example, can be used for time limiting of operation of outputs and automatic functions.
MOON	M out of n is used for supervision and selection of analog values in the case of set error bits or discrepancies between the analog values.
MOVE	Used to move data, for example from one place in the database to another.
MOVERED	Used with unplanned redundancies to copy the input value on the output value.
MUL	Used for multiplication of up to 19 integers or real numbers.
MULDIV	Scaling element for integer data
MUX-I	Selector for up to 19 inputs. Integer address.
MUX-MI	Selector for up to 19 inputs. Integer address and memory.
MUX-MN	Selector for up to 19 inputs. 1-of-N address and memory.
OR	Gives the Boolean sum of up to 19 Boolean values.
OR-A	Gives the Boolean sum of up to 9 Boolean values and up to 5 Boolean products each of a maximum of 9 Boolean values.

PC Element Short Description (Continued)

Element	Description
OR-TRG	Used when the output of an OR gate should be triggered.
OSC-B	Oscillator for a Boolean variable with variable frequency and pulse time.
PB-IN	Used for reading signals from the PROFIBUS.
PB-OUT	Used for writing signals to the PROFIBUS.
PC_AX646	Test AX646 module.
PC_DX610	Test DX610 module.
P-1	Used as an amplifier in feedback systems. The element permits output signal limiting and following.
P_STEP	Used within a sequential control to give the commands step by step for running or stopping of a functional group. It can be used together with GRC.
PCPGM	Module header for program.
PDP	Used to give proportional effect and limited derivation effect. The element permits limiting of the output signal and following.
PI	Used as a standard PI regulator for serial compensation in feedback systems. The element permits limiting of the output signal and following.
PID	PID controller used to create simple control loops on the function group and single-variable level.
PIP	Used to give proportional effect and limited integration effect. The element permits limiting of the output signal and following.
RAMP	Used to limit the rate of change of a signal. The element permits limiting of the output signal and following.
RED_STAT	Used to identify the redundancy state of the system.
REG	Used as a memory element with an optional number of positions.
REG-G	Used to assemble simple variables into one single variable of group type.
RSM	Used for measuring the rotational speed with identification of the direction of the rotation.
SEQ	Module header for a sequence.
SHIFT	Used as a shift register with optional length.
SLAVEM	Module header for a slave. Subordinate to a MASTER.
SQRT	Calculates the square root of a real number. The result is multiplied by a scale factor.
SR	Memory for Boolean variables. The element has one Set and one Reset input.

PC Element Short Description (Continued)

Element	Description
SR-AA	Memory for Boolean variables. The element has And gates with up to 9 inputs on the Set and Reset inputs.
SR-AO	Memory for Boolean variables. The element has one And gate with up to 9 inputs on the Set input and one Or gate with up to 9 inputs on the Reset input.
SR-D	Memory for Boolean variables. The element has one Set and one Reset input and data and clock inputs for clocking in data.
SR-OA	Memory for Boolean variables. The element has one Or gate with up to 9 inputs on the Set input and one And gate with up to 9 inputs on the Reset input.
SR-OO	Memory for Boolean variables. The element has Or gates with up to 9 inputs on the Set and Reset inputs.
STEP	Module header for a step. Subordinate to an SEQ.
SUB	Subtraction of one real or one integer number from another.
SW	Switching element with up to 9 closing channels.
SW-C	Switching elements with up to 9 change over channels.
SW-RED	Specifies for the planned redundancy on which link (HSL) the value of the master will be transmitted. The corresponding HSL- input is switched-through to the output.
SYSL	Calculation of system load.
SYSTIME	Actual time and date.
SYS-CRC	Manage fingerprints.
THRESH-L	Used to determine when more than a given number of Boolean signals are set to 1.
TIMER	Used to generate pulses at up to 9 outputs. The pulse duration is determined for each output by to times within a day.
TOFF	Delay for turning a Boolean signal off.
TON	Delay for turning a Boolean signal on.
TON-RET	Delay for turning a Boolean signal on with accumulating time measurement.
TRAC	Used to select for the redundancy task which one of the redundant controllers has to be tracked to the other controllers.
TRIGG	Provides a pulse during one program cycle when the input signal changes from 0 to 1.
XOR	Used to give a signal when one and only one of the input signals is true.

Product Grouping

This chapter shows which elements are included in the different Advant Controller 160 software packages.

Advant Controller 160 with BASE SW

- ABS, ADD, ADD-MR, AND, AND-O, AND-TRG
- BLOCK, BGET, BSET
- COMP-B, COMP-I, COMP-R, CONTRM, CONV, CONV-AX, CONV-BI, CONV-IB, COUNT
- DIV, DIV-MR
- FIFO, FUNCM
- GRC
- INV
- LIM-N
- MASTER, MAX, MIN, MONO, MONO_UT, MOVE, MOVERED, MUL, MULDIV, MUX-I, MUX-MI, MUX-MN
- OR, OR-AND, OR-TRG, OSC-B
- PCPGM
- PB-IN,PB-OUT
- PC_AX646, PC_DX610, P_STEP
- RED_STAT, REG
- SEQ, SHIFT, SLAVEM, SQRT, SR, SR-AA, SR-AO, SR-D, SR-OA, SR-OO, STEP, SUB, SW, SW-C, SYSL, SYSTIME, SYS-CRC
- THRESH-L, TIMER, TOFF, TON, TON-RET, TRIGG,
- XOR.

Advant Controller 160 with OPT 1 Extended PC Element Set

- CON-PU1, DER, FILT-1P, FUNG-1V, INT, P-1, PDP, PI, PID, PIP, RAMP, REG-G.

Advant Controller 160 with OPT 6 Station Red. + RSM

- MOON, RSM, SW-RED, TRAC.

Execution Times

Name	Execution Time at 33 MHz, typical (μs)	Notes
ABS	7 16 20	C1=I C1=IL C1=R
ADD	3+0.6×C2 3+C2 10.6×C2–3	C1=I C1=IL, C2≥2 C1=R
ADD-MR	5×(1+C1+C2)	–
AND(1)	2+0.8×C1	(0.5+0.4×C1), ( ) = “Inline” time
AND-O	6+2×(C1+C2+C3+C4+C5+C6)	–
AND-TRG	4+0.7×C1	–
BGET	4	–
BLOCK	2	–
BSET	6	–
COMP-B	3.6	–
COMP-I	5	–
COMP-R	18+2.6×(C1+C2)	–
CON-PU1	10 100 10 167	C1=0 At PF×TS Not at PF×TS C1=1 At PF×TS Not at PF×TS
CONTRM	230	–
CONV	14 12 <31	I,IL→ R R→I,IL all other conversions
CONV-AX	10 + 26×C1	–

Execution Times (Continued)

Name	Execution Time at 33 MHz, typical (µs)	Notes
CONV-BI	10+1.3×C3 12+2.3×C3 10+0.6×C3 10+2.3×C3 7+2.6×C3 10+1.3×C3 12+2.3×C3 10+0.6×C3 10+2.3×C3 7+2.6×C3	C1=I, C2=1 (BC→I) C1=I, C2=2 (BCD→I) C1=I, C2=3 (1 of N→I) C1=I, C2=4 (Gray→I) C1=I, C2=5 (16/32BC→I) C1=IL, C2=1 (BC→IL) C1=IL, C2=2 (BCD→IL) C1=IL, C2=3 (1 of N→IL) C1=IL, C2=4 (Gray→IL) C1=IL, C2=5 (16/32BC→IL)
CONV-IB	12+1.3×C3 26+1.3×C3 12+1.3×C3 11+1.6×C3 26+1.6×C3 12+1.3×C3	C1=I, C2=1 (I→BC) C1=I, C2=2 (I→BCD) C1=I, C2=3 (I→1 of N) C1=IL, C2=1 (IL→BC) C1=IL, C2=2 (IL→BCD) C1=IL, C2=3 (IL→1 of N)
COUNT	6	–
DER	60	–
DIV	8 70 25	C1=I C1=IL C1=R
DIV-MR	18+10.6×(C1+C2)	–
FIFO	13+C2	C1=I, B C1=IL, R, T, TR
FILT-1P	73	–
FUNCM	1	–
FUNG-1V	83	–
GRC	211 158	C1=1, C2=0 C1=0, C2=1
INT	43	–
INV	2	–
LIM-N	12+0.6×C2	–

Execution Times (Continued)

Name	Execution Time at 33 MHz, typical (µs)	Notes
MASTER	230	–
MAX	5+1.6×C2 5+2.3×C2 4+3×C2	C1=I C1=IL C1=R
MIN	5+1.6×C2 5+2.3×C2 4+3×C2	C1=I C1=IL C1=R
MONO	7	–
MONO_UT	7	–
MOON	72+68×C2	C2=2…5
MOVE	3+1.3×C2	–
MOVERED	5+2.6×C2	–
MUL	2+2×C2 9.6×C2–4 13×C2–7	C1=I C1=IL, C2≥2 C1=R
MULDIV	6.3	–
MUX-I	6	–
MUX-MI	9+0.8×C2	–
MUX-MN	9+0.8×C2	–
OR(1)	1+0.8×C1	(0.5+0.4×C1), ( ) = “Inline” time
OR-A	6+2×(C1+C2+C3+C4+C5+C6)	–
OR-TRG	4+0.7×C1	–
OSC-B	5	–
PB-IN	31 22 29+2.2×C2	C1=I, IL C1=R C1=B
PB-OUT	30 20 28.2+3.8×C2	C1=I, IL C1=R C1=B
PC_AX646	23+6.6×C1	–

Execution Times (Continued)

Name	Execution Time at 33 MHz, typical (µs)	Notes
PC_DX610	54+27.5×C1	–
P-1	46	–
P_STEP	38	–
PCPGM	230	–
PDP	86	–
PI	80	–
PID	350+150×C1	–
PIP	90	–
RAMP	73	–
RED_STAT	13	–
REG	5+1.2×C2	–
REG-G	7+C2+3×C3+0.6×C4	–
RSM	987	C1=0, C1=1
SEQ	350 (at normal STEP execution) 466 (at STEP-CHANGE)	C3=C4=C5=0 Including STEP
SHIFT	4	–
SLAVEM	1	–
SQRT	56	–
SR(1)	3	(1.5), ( ) = “Inline” time
SR-AA	5+C1+C2	–
SR-AO	5+C1+C2	–
SR-D	4	–
SR-OA	5+C1+C2	–
SR-OO	5+C1+C2	–
STEP	See PC element SEQ	
SUB	3 4 16	C1=I C1=IL C1=R
SW	6+2×C2	–

Execution Times (Continued)

Name	Execution Time at 33 MHz, typical (μs)	Notes
SW-C	5+2.6×C2	–
SW-RED	23	C1=B, I, IL, R, T, TR
SYSL	13+27×C1	–
SYSTIME	70	–
SYS-CRC	110 120	C1=0 C1=1
THRESH-L	7+0.7×C2	–
TIMER	10+5.3×C1	–
TOFF	3	–
TON	3	–
TON-RET	4	–
TRAC	305	–
TRIGG	3	–
XOR(1)	3	(1.2), ( ) = “Inline” time
Inline begin + end	1.7	One-time overhead for an inline sequence!
Type Circuits begin + end	2.1	One-time overhead for a Type Circuit

(1) these elements are available as inline code (see “Space Requirements for inline Coded PC Elements)

The following pages show the memory space requirements for PC elements when all the inputs of each element are connected to the output of another element.

Space Requirements

Name	PC Statement (byte)	Local Data Area (byte)	Notes
ABS	14	2 4	C1=I C1=IL
ADD	10+2×C2	2 4	C1=I C1=IL, R
ADD-MR	12+2×C1+2×C2	–	–

Space Requirements (Continued)

Name	PC Statement (byte)	Local Data Area (byte)	Notes
AND	10+2×C1	2	–
AND-O	14+2×(C1+C2+C3+C4+C5+C6)	2	–
AND-TRG	12+2×C1	4	–
BGET	14	–	–
BLOCK	12	2	–
BSET	18	–	–
COMP-B	12	2	–
COMP-I	18	6	–
COMP-R	18+4×(C1+C2)	8+2×(C1+C2)	–
CON-PU1	54	90	–
CONTRM	38	4+14	–
CONV	14	4	C1=I C1=IL, R, T, TR
CONV-AX	14+10×C1	6+4×C1	–
CONV-BI	24+2×C3	6 8	C1=I C1=IL
CONV-IB	24+2×C3	12+2×C3	–
CONV_AX	14+10×C1	6+4×C1	–
COUNT	30	10 12	C1=I C1=IL
DER	34	44	–
DIV	18	4 8	C1=I C1=IL
DIV-MR	10+2(C1+C2)	2	–
FIFO	24+4×C2	16+2×(C2+C2×C3) 16+4×(C2+C2×C3)	C1=I, B C1=IL, TR, R, T
FILT-1P	30	40	–
FUNCM	6	–	–
FUNG-1V	26	36	–

Space Requirements (Continued)

Name	PC Statement (byte)	Local Data Area (byte)	Notes
GRC	46+6×C2+18×C1	112+32×C1+6×C2	–
INT	32	32	–
INV	10	2	–
LIM-N	24+6×C2	14 20	C1=I C1=IL, R, T, TR
MASTER	38	4+14	–
MAX	14+2×C2	4 6	C1=I C1=IL
MIN	14+2×C2	4 6	C1=I C1=IL
MONO	18	12	–
MONO_UT	18	12	–
MOON	16+6×C2	8+2×C2	–
MOVE	8+4×C2	2×C2 4×C2	C1=B, I C1=IL, R, T, TR
MOVERED	8+4×C2	2×C2 4×C2	C1=B, I C1=IL, R, T, TR
MUL	10+2×C2	2 4	C1=I C1=IL, R
MULDIV	16	4	–
MUX-I	14+2×C2	4 6	C1=B, I C1=IL, R, T, TR
MUX-MI	22+2×C2	6 8	C1=B, I C1=IL, R, T, TR
MUX-MN	20+4×C2	6 8	C1=B, I C1=IL, R, T, TR
OR	10+2×C1	2	–
OR-A	14+2×(C1+C2+C3+C4+C5+C6)	2	–
OR-TRG	12+2×C1	4	–
OSC-B	16	12	–

Space Requirements (Continued)

Name	PC Statement (byte)	Local Data Area (byte)	Notes
PB-IN	12+2×C2	2+2×C2 2+4×C2	C1=B, I C1=IL, R
PB-OUT	8+2×C2	–	–
PC_AX646	18+10×C1	16+10×C1	–
PC_DX610	26+16×C1	48+4×C1	–
P-1	28	14	–
P_STEP	30	56	–
PCPGM	26	2+14	–
PDP	32	48	–
PI	38	36	–
PID	52	36	–
PIP	32	90	–
RAMP	34	52	–
RED_STAT	10	6	–
REG	16+4×C2	2+2×C2 2+4×C2	C1=B, I C1=IL, R, T, TR
REG-G	28+2×C2+2×C3	6+2×C5 6+4×C5	C1=B, I C1=IL, R, T, TR
RSM	32	46	–
SEQ	20+14×C3+8×C4+4×*k* (*k* = number of STEP's)	200+14+2×C3+8×C4	–
SHIFT	24	2+2×C2 2+4×C2	C1=B, I C1=IL, R, T, TR
SLAVEM	6	–	–
SQRT	14	6	–
SR	12	2	–
SR-AA	10+2×(C1+C2+C3+C4+C5+C6)	–	–
SR-AO	10+2×(C1+C2+C3+C4+C5+C6)	–	–
SR-D	18	4	–

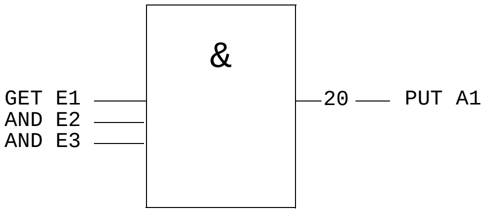
Space Requirements (Continued)

Name	PC Statement (byte)	Local Data Area (byte)	Notes
SR-OA	10+2×(C1+C2+C3+C4+C5+C6)	–	–
SR-OO	10+2×(C1+C2+C3+C4+C5+C6)	–	–
STEP	22+8×C1+4×C2+4×C3+2×C4	2+6×C1	–
SUB	14	2 4	C1=I C1=IL, R
SW	10+6×C2	2×C2 4×C2	C1=B, I C1=IL, R, T, TR
SW-C	12+2×C2	2×C2 4×C2	C1=B, I C1=IL, R, T, TR
SW-RED	14	2 4	C1=B, I C1=IL, R, T, TR
SYSL	10+8×C1	8+6×C1	–
SYSTIME	22	16	–
SYS-CRC	28+2×C1	8+28×C1	–
THRESH-L	10+2×C2	2	–
TIMER	12+12×C1	12+2×C1	–
TOFF	16	12	–
TON	16	12	–
TON-RET	18	12	–
TRAC	28	30	–
TRIGG	12	4	–
XOR	12	2	–
Type Circuit begin + end	8+4	–	One-time overhead for a Type Circuit

Space Requirements for Inline Coded PC Elements

Name	PC Statement (byte per terminal)	Local Data Area (byte)	Notes
AND	6	2	–
OR	6	2	–
SR	10	2	–
XOR	8	2	–

.



Example for assignment of inline operations to PC element terminals

Generation of Inline Code

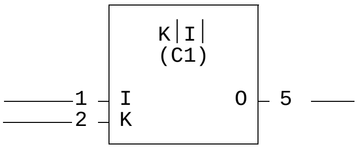
Inline code is automatically generated by the Advant Station 120 Engineering Station if at least two PC elements of types AND, OR, SR, or XOR follow another order. From the user’s viewpoint there is no difference between a normal PC element and the corresponding inline element.

Inline PC elements are used for speed optimization of binary operations. INV can be an additional attribute of a GET-, AND-, OR-, SR-, or XOR-terminal.

Absolute Value ABS

Summary

ABS (**ABS**olute value) is used to form the absolute value of an integer or a real number. The absolute value can be multiplied with an optional value.



PC element ABS

Call ABS (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	I	IC*1*	**I**nput. Input for number, the absolute value of which is to be obtained.
2	K	O*C1*	Input for multiplication factor K.
5	O	O*C1*	**O**utput. Output for the product of the value at input K and the absolute value of the value at the input I.

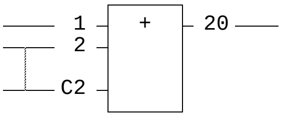
Function

The value at input K is multiplied by the absolute value of the input I and the result is stored at the output O.

Adder ADD

Summary

ADD is used for addition of an optional number of integers or real numbers



PC element ADD

Call ADD (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R
C2	Number of inputs	2…19

Terminal Description

No	Name	Type	Description
1	–	I*C1*	Input for addend.
2	–	I*C1*	Input for addend.
: C2	–	I*C1*	Input for addend.
20	–	I*C1*	Output for sum.

Function

The values at the inputs 1…C2 are added and the sum is stored at the output 20.

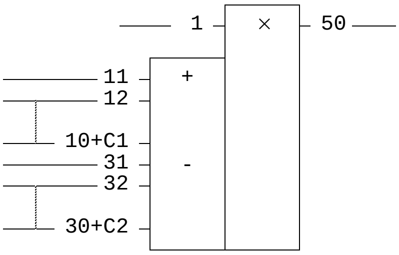
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Adder with Multiplied Output ADD-MR

Summary

ADD-MR (**ADD**er **-** with **M**ultiplier **R**eal numbers) is used for addition of an optional number of real numbers.
The numbers are added in two groups, each of a maximum of 19 numbers, after which the second group is subtracted from the first. The result is multiplied with a real number.



PC element ADD-MR

Call ADD-MR (C1, C2)

Parameter	Significance	Permissible values
C1	Number of positive inputs in the addition element.	1…19
C2	Number of negative inputs in the addition element.	1…19

Terminal Description

No	Name	Type	Description
1	–	IR	Input for multiplication factor.
11	–	IR	Input for addend with positive weight factor.
12	–	IR	Input for addend with positive weight factor.
: 10+C1	–	IR	Input for addend with positive weight factor.
31	–	IR	Input for addend with negative weight factor.
32	–	IR	Input for addend with negative weight factor.
: 30+C2	–	IR	Input for addend with negative weight factor.
50	–	OR	Output for result.

Function

The sum of the real numbers at the inputs 31…30 + *C2* is subtracted from the sum of the inputs 11…10 + *C1*, after which the result is multiplied with the value at input 1. The result is stored at the output 50.

Overflow

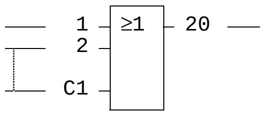
If the maximum positive or negative real value is exceeded, the output is limited to the greatest or lowest representable value respectively.

And Gate/Or Gate

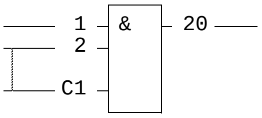
AND, OR

Summary

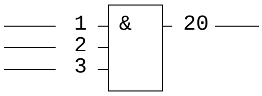
OR and AND elements are used to form general combinatorial expressions with Boolean variables.
The elements can have a maximum of 19 inputs.



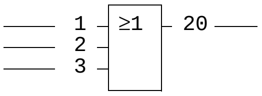
PC element OR



PC element AND



PC element AND (3)



PC element OR (3)

Call OR (C1) AND (C1)

Parameter	Significance	Permissible values
C1	Number of inputs	2…19

Terminal Description

No	Name	Type	Description
1	–	IB	Input.
2	–	IB	Input.
: C1	–	IB	Input.
20	–	OB	Output.

Function OR Element

The output signal is set if any of the inputs are set. See the table below.

Table OR

1	2	3	20
0	0	0	0
0	0	1	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1

Table OR (Continued)

1	2	3	20
1	1	1	1

Function AND Element

The output signal is set to 1 if all of the inputs are 1. See the table below.

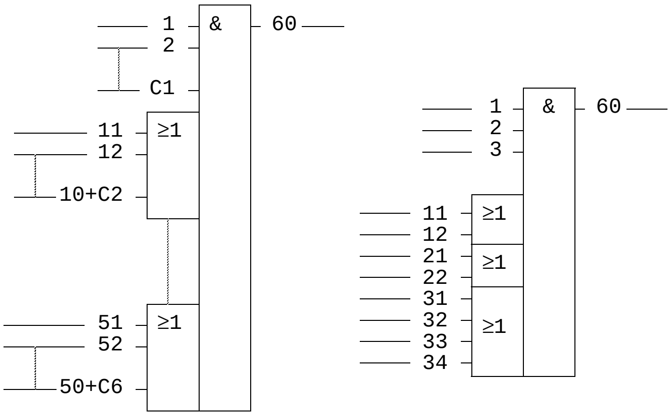
Table AND

1	2	3	20
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

And Gate With Or Inputs AND-O

Summary

AND-O (**AND -** with **O**r gates on the inputs) is used to form general combinatorial expressions with Boolean variables.



PC element AND-O

Call AND-O (C1, C2, C3…C*n*)

Parameter	Significance	Permissible values
C1	Number of inputs in the AND part of the element.	0…9
C2	Number of inputs in the first OR gate of the element.	0…9
C3	Number of inputs in the second OR gate of the element.	0…9
: C*n*(1)	Number of inputs in the last OR gate of the element.	0…9

(1) $1 \le n \le 6$ and $1 \le C1+C2+\ldots+Cn \le 40$

Terminal Description

No	Name	Type	Description
1	–	IB	Input direct to the AND gate in the element.
2	–	IB	Input direct to the AND gate in the element.
: C1	–	IB	Input direct to the AND gate in the element.
11	–	IB	Input to the first OR gate in the element.
12	–	IB	Input to the first OR gate in the element.
: 10+C2	–	IB	Input to the first OR gate in the element.
21	–	IB	Input to the second OR gate in the element.

Terminal Description (Continued)

No	Name	Type	Description
22	–	IB	Input to the second OR gate in the element.
: 20+C3	–	IB	Input to the second OR gate in the element.
: 10x(*n*-1)+1	–	IB	Input to the last OR gate in the element.
10x(*n*-1)+2	–	IB	Input to the last OR gate in the element.
: 10x(*n*-1)+C*n*	–	IB	Input to the last OR gate in the element.
60	–	IB	Output.

Function

More rapid execution than if separate elements are used can be obtained by using assembled elements such as AND-O. The inputs need then only be tested until the value of the output can be determined. If input 1 is 0, the output can be reset irrespective of the other inputs. See the table below.

Table AND-O (3, 3, 3)

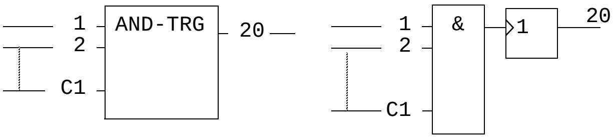
1	2	3	11	12	13	21	22	23	60
0	×(1)	×	×	×	×	×	×	×	0
1	0	×	×	×	×	×	×	×	0
1	0	0	×	×	×	×	×	×	0
1	1	1	0	0	0	×	×	×	0
1	1	1	0	0	1	0	0	0	0
1	1	1	0	0	1	0	0	1	1
1	1	1	0	0	1	0	1	×	1
1	1	1	0	0	1	1	×	×	1
1	1	1	0	1	×	0	0	0	0
1	1	1	0	1	×	0	0	1	1
1	1	1	0	1	×	0	1	×	1
1	1	1	0	1	×	1	×	×	1
1	1	1	1	×	×	0	0	0	0
1	1	1	1	×	×	0	0	1	1
1	1	1	1	×	×	0	1	×	1
1	1	1	1	×	×	1	×	×	1

(1) × indicates that the input has no effect on the value of the output or the execution time.

And Gate With Trigger on Outputs AND-TRG

Summary

AND-TRG (**AND** - with **TR**i**G**ger on the output) is used when the output of an AND gate should be triggered.



PC element AND-TRG

Call AND-TRG (C1)

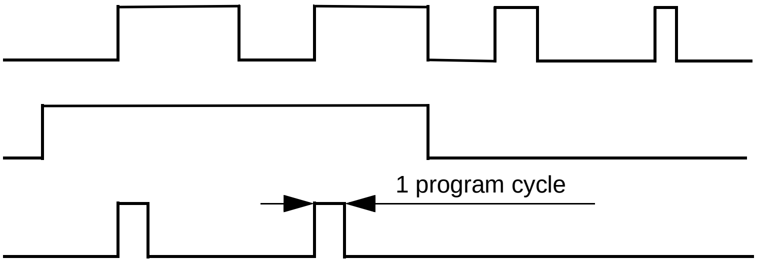
Parameter	Significance	Permissible values
C1	Number of AND inputs	2…19

Terminal Description

No	Name	Type	Description
1	–	IB	Input to the AND gate.
2	–	IB	Input to the AND gate.
: C1	–	IB	Input to the AND gate.
20	–	OB	Output.

Function

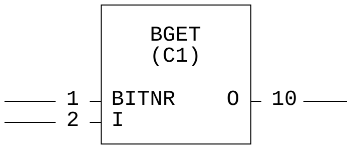
When the result of the AND gate is set (to 1) the output signal is also set. The output signal is cleared during the next program cycle, irrespective of the result of the AND gate. For the output signal to be set, the result of AND gate must have been 0 during one program cycle. The output signal from the element may only be used within its own execution unit as detection of it is otherwise not certain.



Read one Bit BGET

Summary

BGET is used to read one selected bit from an integer or long integer value. The integer usually contains packed boolean data.



PC element

Call BGET (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description	Values
1	BITNR	II	The selected **BIT N**umbe**R** to read.	0…15 (C1 = I) or 0…31 (C1 = IL)
2	I	I*C1*	**I**nput. Data input.	
10	O	OB	**O**utput. Status of the bit selected.	

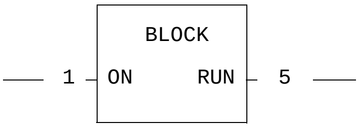
Function

The bit value of input I, specified by BITNR, is loaded to output O. If BITNR is not the range 0…15 (C1 = I) or 0…31 (C1 = IL), then output O is set to 0.

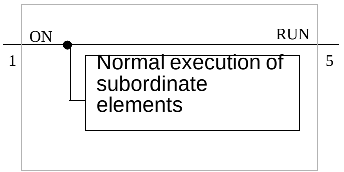
Block Header BLOCK

Summary

Block header BLOCK is used to connect and disconnect the execution of several PC elements.



PC element BLOCK



Function diagram

Call BLOCK

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) for normal execution.
5	RUN	OB	**RUN** indicates execution of the elements in the block.

Function

BLOCK is used to permit the connection and disconnection of a number of PC elements in a control module, slave or sequence step. A block may not contain structure elements.

Execution

For normal execution of a block, the superior execution unit must be under normal execution, and the input ON of the block header must be set to 1. The execution of the block is then performed according to the conditions of the superior execution unit. If the ON input is set to 0 after the block has been executed, the calculated data remains until the next time the block is executed. If the superior execution unit is being executed in the reset mode, the elements in the block will also be executed in that mode, irrespective of the state of the ON input.

Blocking

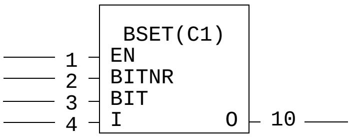
Blocking requires the ON input to be set to 0. At reset the elements in the block will stop executing, but the latest value at each output will be read out to the database unchanged. To reset the outputs, the superior execution unit must be executed in the reset mode. The superior execution unit will then read out 0-values to the database. Blocking from a service aid is performed independently of the ON input of the block in accordance with the status of the execution unit which includes the block.
Note that time delays under BLOCK, for example TOFF, will increase the delay time with the time of blocking.

he output RUN is set only if normal execution is in progress, that is the superior execution unit is under normal execution and the ON input of the block is set.

Set one Bit BSET

Summary

BSET is used to set one selected bit of an integer or long integer value.The integer usually contains packed boolean data.



PC element BSET

Call BSET (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description	Values
1	EN	IB	**EN**able. If EN is reset (to 0), then the current value I is loaded unchanged to output O.	
2	BITNR	II	Input for selecting the **BIT N**umbe**R** to set.	0…15 (C1 = I) or 0…31 (C1 = IL)
3	BIT	IB	Input. Value of the **BIT.**	0…1
4	I	I*C1*	**I**nput. Data input.	
10	O	OB	**O**utput. Status of the bit selected.	

Function

If the value of input EN is set (to 1) then the data bit, determined by BITNR, is replaced by the value of the input bit BIT and the result is loaded to output O.

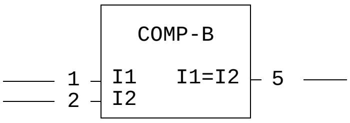
RUN

The output RUN is set only if normal execution is in progress, that is the superior execution unit is under normal execution and the ON input of the block is set.

Comparator for Boolean Values COMP-B

Summary

COMP-B (**COMP**arator **- B**oolean) is used to compare two integers.



PC element COMP-B

Call COMP-B

Terminal Description

No	Name	Type	Description
1	I1	IB	**I**nput **1**. Input, the value of which is to be compared with input I2.
2	I2	IB	**I**nput **2**. Input, the value of which is to be compared with input I1.
5	I1=I2	OB	**I**nput **1** = **I**nput **2**.

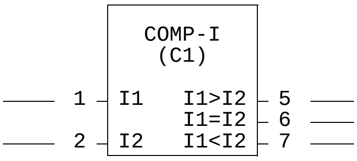
Function

The values at the two inputs are compared and the result of the comparison can be read at the output. The output signal is 0 if the input signals are different and 1 if they are equal.

Comparator for Integers COMP-I

Summary

COMP-I (**COMP**arator **- I**nteger) is used to compare two integers.



PC element COMP-I

Call COMP-I (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description
1	I1	IC1	**I**nput **1**. Input, the value of which is to be compared with input I2.
2	I2	IC1	**I**nput **2**. Input, the value of which is to be compared with input I1.
5	I1>I2	OB	**I**nput **1** > **I**nput **2**. Output which is set if the value at the input I1 is greater than the value at input I2.
6	I1=I2	OB	**I**nput **1** = **I**nput **2**. Output which is set if the value at the input I1 is equal to the value at input I2.
7	I1<I2	OB	**I**nput **1** < **I**nput **2**. Output which is set if the value at the input I1 is less than the value at input I2.

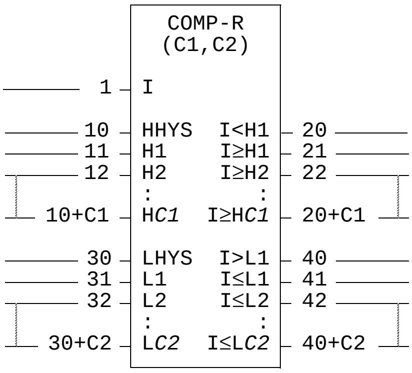
Function

The values at the two inputs are compared and the result of the comparison can be read at the outputs I1<I2, I1=I2 and I1>I2. At startup the output I1=I2 is set before the first execution independent of the values of the inputs I1 and I2.

Comparator for Real Numbers COMP-R

Summary

COMP-R (**COMP**arator **- R**eal) is used for limit value monitoring of real numbers with relation to several limits.



PC element COMP-R

Call COMP-R (C1, C2)

Parameter	Significance	Permissible values
C1	Number of limit values for high level	1…9
C2	Number of limit values for low level	1…9

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input, the value of which is to be compared with limit values H1…H*C1* and L1…L*C2*.
10	HHYS	IR	**H**igh **HYS**teresis. Hysteresis common for all H values. The hysteresis is specified as the difference between the I-value at which the H outputs are set and the value at which they are reset.
11	H1	IR	**H**igh **1**. Upper limit value 1.
12	H2	IR	**H**igh **2**. Upper limit value 2.
: 10+C1	H*C1*	IR	**H**igh ***C1***. Upper limit value C1.
20	I<H1	OB	**I**nput < **H1**. Output which is set if I is less than H1.
21	I≥H1	OB	**I**nput ≥ **H1**. Output which is set if I is greater than or equal to H1.
22	I≥H2	OB	**I**nput ≥ **H2**. Output which is set if I is greater than or equal to H2.
: 20+C1	I≥H*C1*	OB	**I**nput ≥ **H***C1*. Output which is set if I is greater than or equal to HC1.
30	LHYS	IR	**L**ow **HYS**teresis. Hysteresis common for all L values. The hysteresis is given as the difference between the I- value at which the L outputs are reset and the value at which they are set.

Terminal Description (Continued)

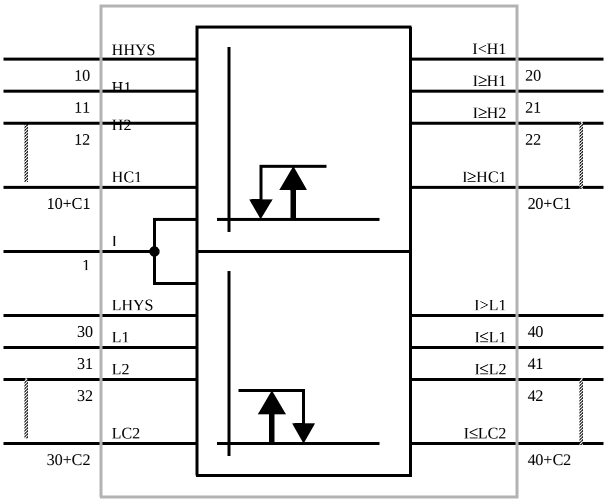
No	Name	Type	Description
31	L1	IR	**L**ow **1**. Lower limit value 1.
32	L2	IR	**L**ow **2**. Lower limit value 2.
: 30+C2	L*C2*	IR	**L**ow ***C2***. Lower limit value C2.
40	I>L1	OB	**I**nput > **L1**. Output which is set if I is greater than L1.
41	I≤L1	OB	**I**nput ≤ **L1**. Output which is set if I is less than or equal to L1.
42	I≤L2	OB	**I**nput ≤ **L2**. Output which is set if I is less than or equal to L2.
: 40+*C2*	I≤L*C2*	OB	**I**nput ≤ **L*C2***. Output which is set if I is less than or equal to LC2.

Function

Input signal I is compared with the limit value specified at the inputs H1…H*C1* and L1…L*C2*. For upper limits, the output for the different limits will be set when the input I becomes equal to or greater than the limit value. For lower limits, the output is set when I becomes less than or equal to the limit value. Limits can be set optionally within the interval 9.2×10^{18}…-0.1×10^{18}. The lowest representable positive and negative values are 5.0×10^{-20} and -5.0×10^{-20}. The element put outputs 20 (I<H1) and 40 (I>L1) to 1 before the first execution independent of the input I.

Hysteresis

The input HHYS gives the hysteresis for all limits at high level and LHYS for all limits at low level. The hysteresis is given as the difference between the I values for which the different outputs are set and the values for which they are reset. At high level the outputs will be reset when I becomes lower than the limit minus the hysteresis HHYS. At low level the outputs will be reset when I becomes greater than the limit value plus the hysteresis LHYS.

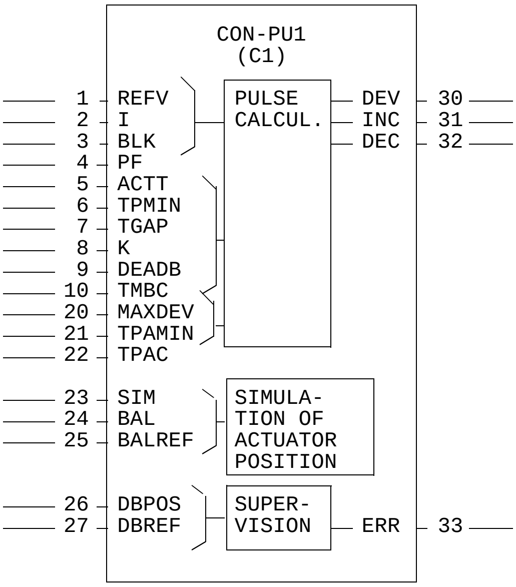


Function diagram

Three State Controller CON-PU1

Summary

CON-PU1 (**CON**troller **- PU**lsed **1**) is a three state controller intended to control, by means of pulses (increase/decrease), a process via an integrating actuator such as a shifting motor of a solenoid valve. CON-PU1 can be combined with, for example, a PI regulator for process control. If only pulse generation is desired, a simplified function can be chosen with a call parameter.



PC element CON-PU1

Call CON-PU1 (C1)

Parameter	Significance	Permissible values
C1	Selection of extended or ordinary function. The value 0 gives ordinary function.	0…1

Terminal Description

No	Name	Type	Description
1	REFV	IR	**REF**erence **V**alue. Input for setpoint.
2	I	IR	**I**nput value. Input for measured value.
3	BLK	IB	**BL**oc**K** pulse calculation.
4	PF	II	**P**ulse **F**requency Factor. Minimum pulse cycle time = PF × TS.
5	ACTT	ITR	**ACT**uator **T**ime. The delay time interval of the actuator.
6	TPMIN	ITR	**T**ime **P**ulse **MIN**imum. Min pulse length for excitation of the motor.
7	TGAP	ITR	**T**ime **GAP**. Compensation for play.
8	K	IR	Pulse length factor. Pulse = (K×DEV×ACTT) /100+TGAP+TPMIN.

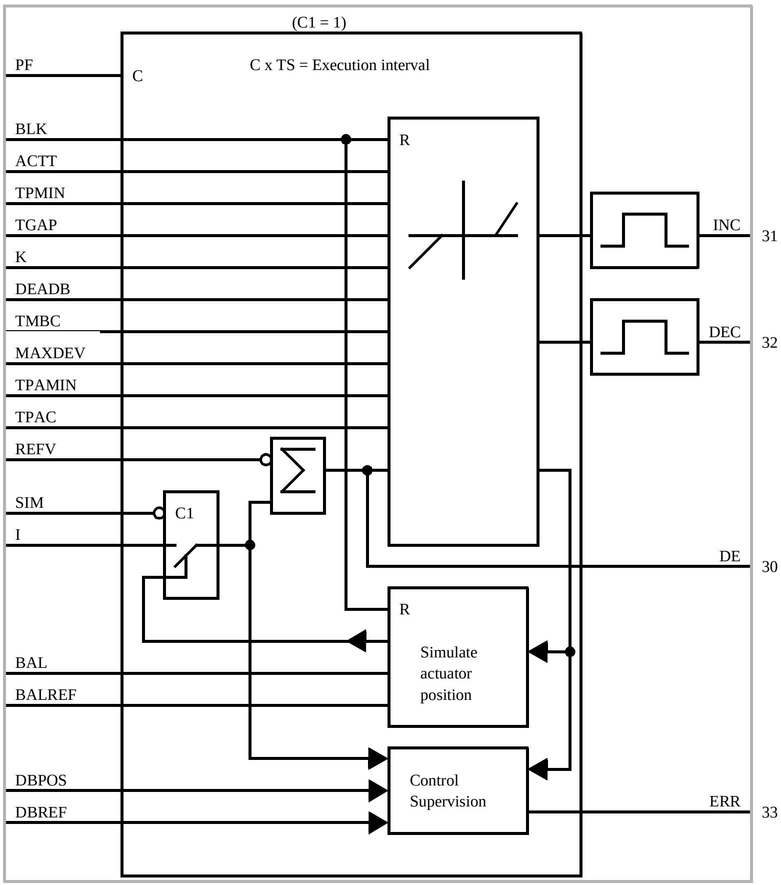
Terminal Description (Continued)

No	Name	Type	Description
9	DEADB	IR	**DEADB**and. $\|DEV\| \geq DEADB$ if a pulse is to be given.
10	TMBC	ITR	**T**ime **M**inimum **B**etween **C**hange. Minimum pause time after DEV having changed sign.
20(1)	MAXDEV	IR	**MAX**imum **DEV**iation. Maximum deviation for pulse length calculation. This value does not affect output DEV.
21(1)	TPAMIN	ITR	**T**ime **PA**use **MIN**imum. Minimum pause time.
22(1)	TPAC	IR	**T**ime **PA**use **C**onstant. Pause Time factor. Pulse time = TPAMIN + TPAC × Pulse duration.
23(1)	SIM	IB	**SIM**ulate actuator position. Input for activation of simulation of actuator position.
24(1)	BAL	IB	**BAL**ance. Input for activation of tracking.
25(1)	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when tracking.
26(1)	DBPOS	IR	**D**ead**B**and for **POS**ition change. Deadband for supervision of position change of actuator.
27(1)	DBREF	IR	**D**ead**B**and **REF**erence for position change alarm. When $\left\| \int_{t0}^{tn} DEV \times K\,dt \right\| > DBPOS$ a check is made to determine if $\|I_{tn} - I_{t0}\| > DBREF$ and if so, the error signal ERR is set.
30	DEV	OR	Position **DEV**iation. DEV = I – REFV.
31	INC	OB	**INC**rease. Output for increase of actuator position.
32	DEC	OB	**DEC**rease. Output for decrease of actuator position.
33(1)	ERR	OB	**ERR**or. Actuator position has not followed the change commanded.

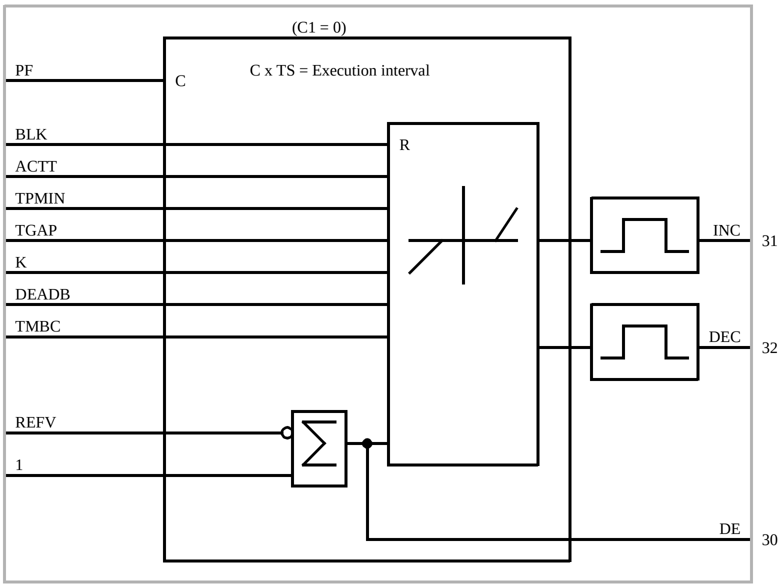
(1) Only used with extended function, (C1=1)

Function

The three state controller CON-PU1 is intended to control, by means of pulses at the outputs INC and DEC (increase/decrease respectively), a process via for example a shifting motor or a solenoid valve. With the call parameter C1 can be selected ordinary (C1 = 0) or extended (C1 = 1) function. The following inputs should be given in percent (0…100): REFV, I, DEADB, MAXDEV, BALREF, DBPOS, DBREF.



Function diagram for extended function (C1=1)



Function diagram for ordinary function (C1=0)

Extended Function (C1 = 1)

The feedback can be obtained from the actuator or the element can simulate the actuator position internally. The effective pulse duration is proportional to the control error and is calculated as follows:

$PL_{eff} = |DEV \times K \times ACTT|/100$, where:

DEV = Control deviation = I–REFV (in percent)

K = Pulse length factor

ACTT = Actuator delay time interval in seconds

To permit gradual correction of a large control error, $|DEV|$, is limited to MAXDEV. The actual time pulse duration is calculated as:

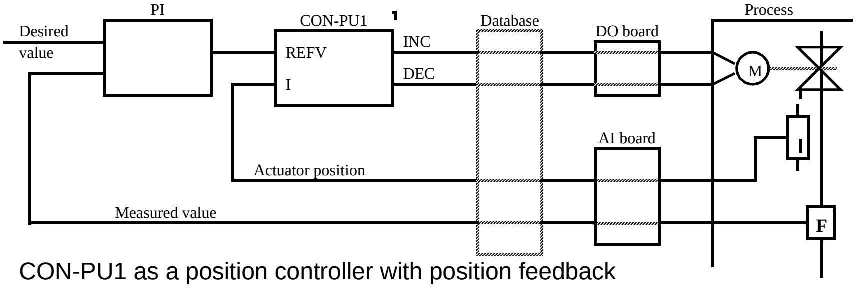
$PL = (|DEV \times K \times ACTT|)/100 + TGAP + TPMIN$ where:

TPMIN = Min pulse duration for excitation of the shifting motor or solenoid valve. This is the shortest pulse length which affects the actuator.

TGAP = Play compensation. Added to the first pulse after a polarity change.

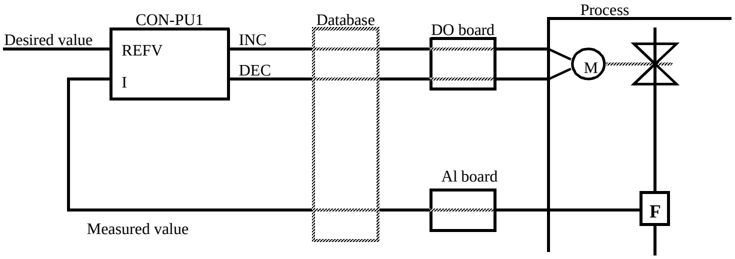
For control errors less than a deadband (DEADB, in percent) the correction is considered to be complete and no further pulse is calculated. TMBC is a delay time that gives the shortest pause time after a change of the pulse outputs INC/DEC.

Calculation of a new pulse duration is activated at the interval PF×TS where TS is the sample time of the function, that is the cycle time of the element.

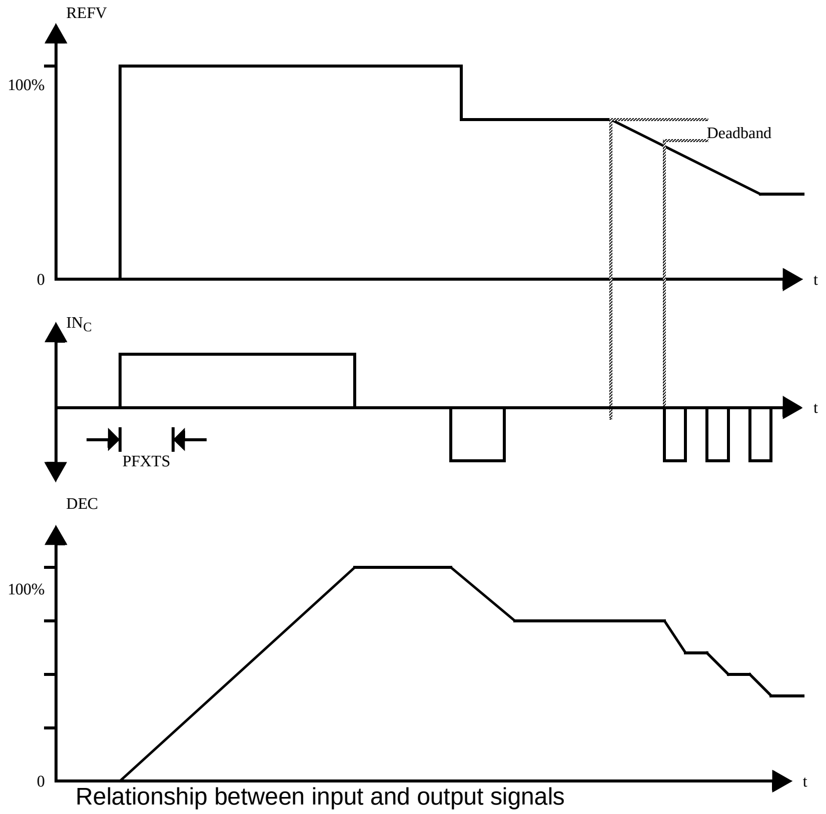


CON-PU1 as a position controller with position feedback

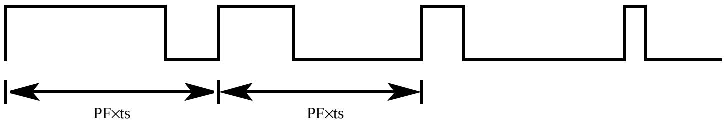
The least permitted pause time between two pulses is determined however by the parameters TPAMIN and TPAC so that the Pause time = TPAMIN+TPAC×PL where PL is the pulse duration calculated most recently. These possibilities of setting the pause time are normally only used when CON-PU1 is used as a stand alone regulator and provide great flexibility in trimming the regulator. By setting PF×TS relatively large and TPAMIN = TPAC = 0, a constant pulse frequency is obtained. To obtain a constant pause time between the pulses, TPAC is set to 0 and TPAMIN to the required pause time. This is necessary when controlling processes with long time constants.



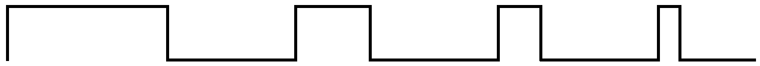
CON-PU1 as a stand alone controller



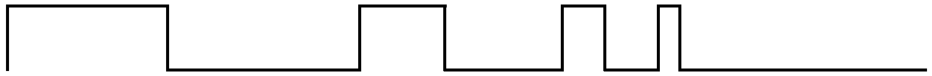
Relationship between input and output signals



Pause time = TPAMIN + TPAC x PL, TPAMN =TPAC = 0



Pause time = TPAMIN + TPAC x PL, TPAC = 0



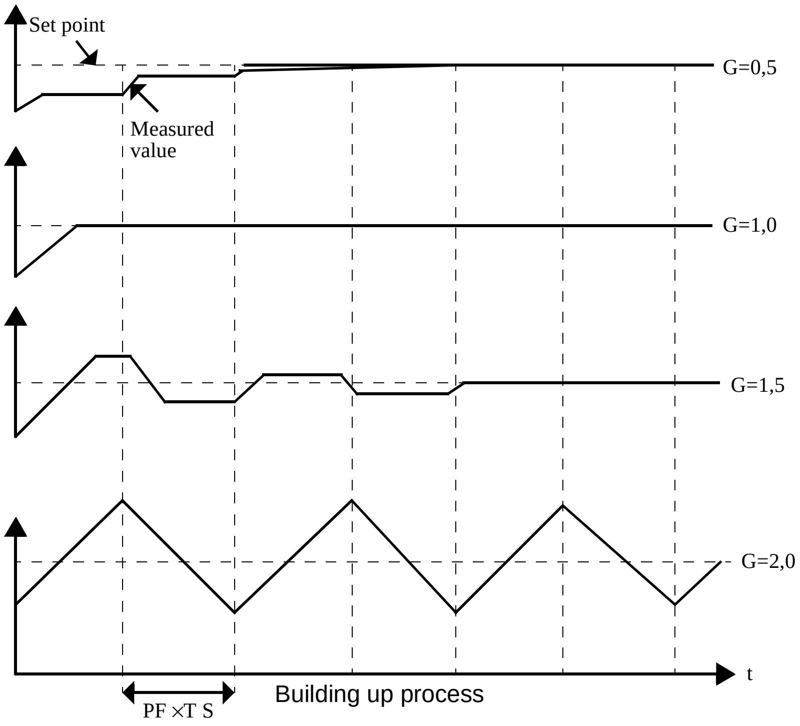
Pause time = TPAMIN + TPAC × PL. TPAMIN and TPAC > > 0

In processes with long delay times, pause times ≥ the delay time are selected.

The pause time can be made to be dependent on the duration of the latest pulse calculated with the parameter TPAC. This function is used when a rapid control of small errors is required but when limits on the speed of change in the process response require long pause times with large changes.

When CON-PU1 is used as a position regulator for a motor actuator, different building up processes are obtained depending on a parameter G, this being in principle the relationship between the real data of the actuator and the parameters set in CON-PU1.

$G = (k \times ACTT)/ACTT_v + 100 \times (TGAP - TGAP_v + TPMIN - TPMIN_v)/(ACTT_v \times |DEV|)$, where $TGAP_v$ is the actuator play., $TPMIN_v$ the least pulse length to which the actuator reacts and $ACTT_v$ the time delay time interval of the actuator. The figure below shows the building up process for different G, where only $(k \times ACTT)/ACTT_v$ has been included. The system is unstable when G≥2.



Simulation of Actuator Position

When it is not possible to feed back the actuator position to the element, the position can be simulated internally in the element by integrating DEV × K × ACTT. The simulated value is limited to between 0 and 100%. When BAL is set, BALREF is used as an integrator value. This is used, for example, with a limit switch on the actuator to set the integrator to a known value or forced control when manually controlling the process. With simulation, the preceding regulator should have an output signal with greater range than 0 to 100% (for example: -100 to 200%).

Supervision

When $\sum_{n=1}^{N} |DEV_n \times K_n| > DBPOS$, a test is made to determine if $|I_N - I_0|$ >DBREF. DEV_n is the deviation at the beginning of pulse n, that is the deviation which is used for calculating the pulse length. I_0 is the feedback signal before the first pulse, I_N is the feedback signal after the N^{th} pulse.

This means that the supervision sums DEV × K for each pulse, and when the sum has exceeded the value DBPOS a check is made (after concluded pulse time) that the feedback signal has changed by at least the value DBREF. If the feedback signal has not changed enough, ERR is set. After the check, the sum is cleared.

The sum is cleared, and ERR is reset, when DEV changes sign or when DBPOS is not exceeded after 9 pulses. Theoretically you can set DBREF = DBPOS. This means that the actuator follows the controller without delay. In practice, you will have to set DBREF < DBPOS.

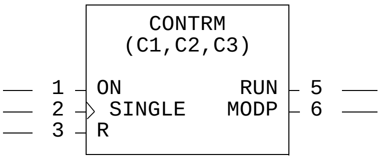
Ordinary Function (C1 = 0)

With ordinary function only pulse generation according to $PL = (|DEV| \times KI \times ACT)/100+TPMIN+TGAP$ is performed.
If $|DEV| < DEADB$ no pulse is obtained.

Control Module Header CONTRM

Summary

CONTRM (**CONTR**ol **M**odule Header) is used for structuring and execution control of control modules. Function modules, blocks and elements can be included in a control module.



PC element CONTRM

Call CONTRM (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Cyclicity in ms.	Allowed values are between 2 ms and 20,000 ms in steps of 2 ms.
C2	Place in the cycle time table and/or scheduling strategy.	1…251: Place in cycle time table (cyclic activation). 252: Activation at initialization time after power down. 255: Event driven by onboard I/O.
C3	Currently not used.	–

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) with normal execution.
2	SINGLE	IB	If input ON and R are 0 the elements in the control module can be executed once by setting **SINGLE**.
3	R	IB	**R**eset. Input for clearing of the elements in the control module.
5	RUN	OB	**RUN** indicates that the elements in the control module are executed.
6	MODP	OB	**MOD**ify **P**ermission. Output that indicates if the module may be modified from service aids.

Function

The control module header CONTRM is a superior execution controlling element for a control module. The call parameter C1 is used to specify how often the control module is to be executed and call parameter C2 is used to specify the place in the cycle time table on which the execution sequence of execution units with the same cycle time is based. When and how the control module is to be executed is determined with the control inputs ON, SINGLE and R.

Normal Execution

The input ON is to be set with normal execution and the control module is then executed in accordance with the call parameters C1 and C2. If the ON input is reset after execution of the control module, the calculated data remains until the next time the control module is executed. The ON input overrides the SINGLE input, that is if ON is set SINGLE has no effect. The execution is performed in three steps:

1. Entry of variables from the data base and from modules with higher priority.
2. Element execution.
3. Reading of variables to the database and to modules with higher priority.

SINGLE Execution

If the SINGLE input is set when the ON input is reset, the control module is executed once in accordance with normal execution.

Clearing

If input R is set, the control module is executed in the reset mode. This means that all outputs of the elements within the control module are given original values which in most cases are the “0” value of the data. Reading to the data base is performed with data value equal 0 for all data types. Input R overrides the inputs ON and SINGLE.

RUN

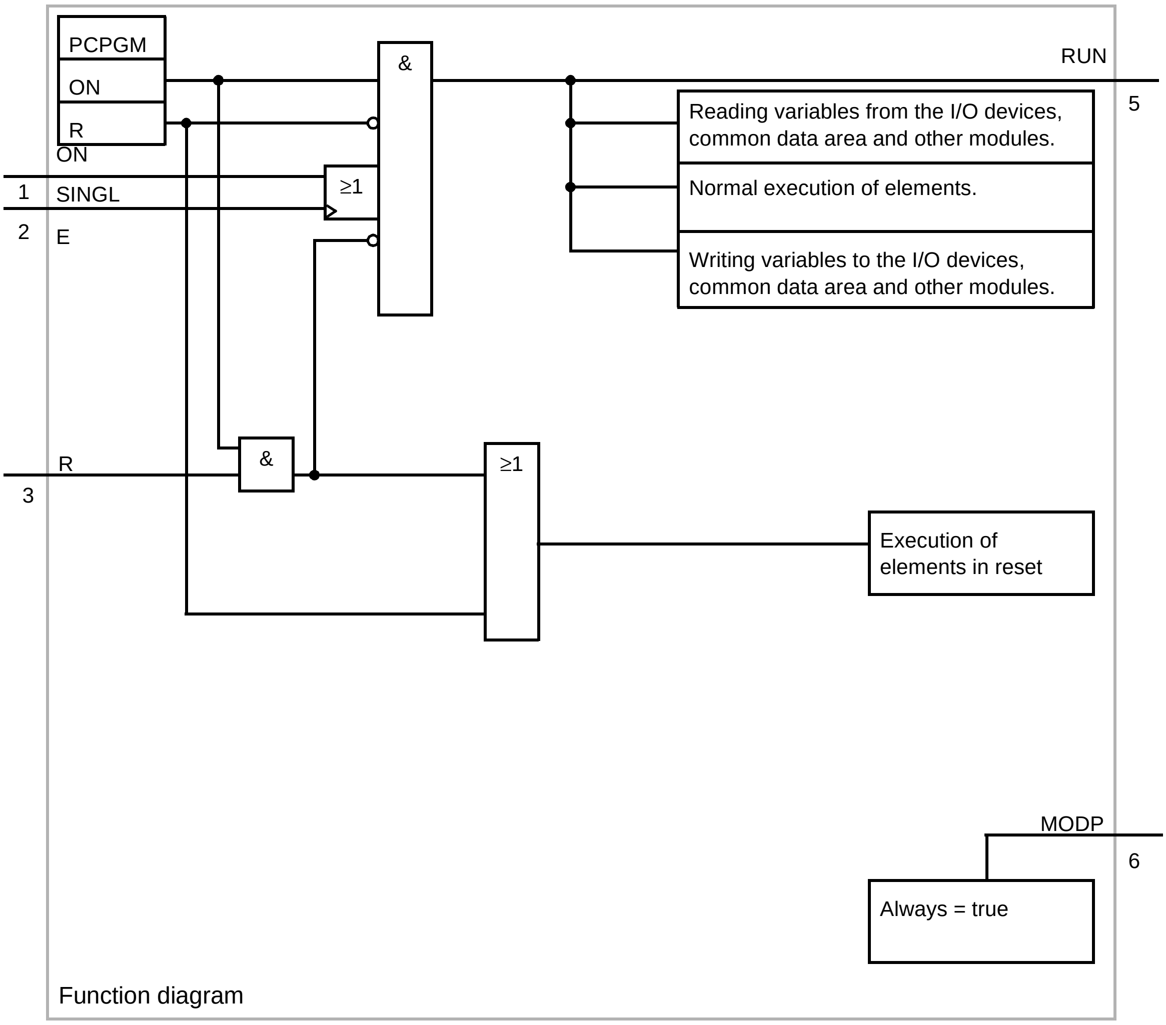
Output RUN is set only if normal execution is in progress, that is if ON is set, or if the control module is being SINGLE executed. With SINGLE execution, the RUN output is set during one cycle only.

MODP

Is always true.

Effects from PCPGM

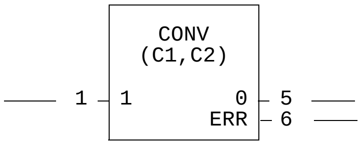
ON and the R inputs and the blocking functions on the program header override the inputs on the control module header. The input ON on PCPGM must be set to permit normal execution or reset execution of the elements in the control module.


PCPGM
ON
R
&
RUN
5
Reading variables from the I/O devices, common data area and other modules.
Normal execution of elements.
Writing variables to the I/O devices, common data area and other modules.
ON
1
SINGL
≥1
2
E
R
&
3
≥1
Execution of elements in reset
MODP
6
Always = true
Function diagram

Code Converter CONV

Summary

CONV (**CONV**erter) converts integer, real, time and time-real data from one representation to each other.



PC element CONV

Call CONV (C1, C2)

Parameter	Significance	Permissible values
C1	Data type of input	I, IL, R, T, TR
C2	Data type of output	I, IL, R, T, TR

Terminal Description

No	Name	Type	Description
1	I	I*C1*	**I**nput. Input for data to be converted.
5	O	O*C2*	**O**utput. Output for converted data.
6	ERR	OB	**ERR**or. Output which is set (to 1) if a limit is reached at output O.

Function

Data at input I with data type in accordance with call parameter C1 is converted to data with data type according to call parameter C2 at the output O. When conversion is to the data types I and IL (integer) and T (time), overflow can occur at the max or the min. limit values. The error signal output ERR is set when overflow occurs.

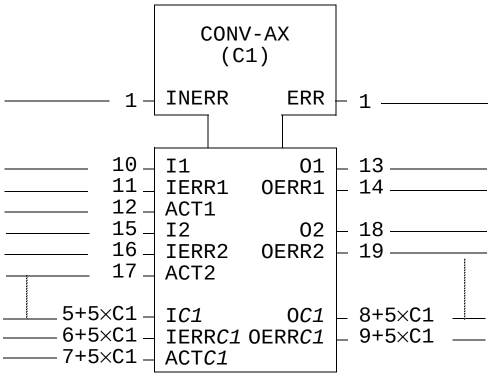
Rounding Off

When converting from real numbers to integers, the number is rounded up when the decimal is 0.5 or greater, and down when the decimal is less than 0.5. When converting from T to R, I or IL, the value given is the number of entire seconds from midnight. When converting from R, I, or IL to T, the input value is treated as the number of entire seconds from midnight.

Convert AX646 Analog Signals to Binary CONV-AX

Summary

CONV-AX converts analog input values of an AX646 module to digital input values. Analog values 4 to 8 mA are converted to “0”, values 12 to 20 mA are converted to “1”.



PC element CONV-AX

Call CONV-AX (C1)

Parameter	Significance	Permissible values
C1	Number of channels	1…16

Terminal Description

No	Name	Type	Description
1	INERR	IB	**ERR**or flag for **IN**put.
2	ERR	OB	**ERR**or flag for output.
10	I1	IR	Analog **I**nput value **1**.
11	IERR1	IB	**ERR**or bit for Analog **I**nput value **1**.
12	ACT1	IB	Binary Input **1** is **ACT**iv.
: 5+5×C1	I*C1*	IR	Analog **I**nput value ***C1***.
6+5×C1	IERR*C1*	IB	**ERR**or bit for Analog **I**nput value ***C1***.
7+5×C1	ACT*C1*	IB	Binary Input ***C1*** is **ACT**iv.
13	O1	OB	Binary converted **O**utput value of analog input value **1**.
14	OERR1	OB	**ERR**or bit for binary **O**utput value **1**.
: 8+5×C1	O*C1*	OB	Binary converted **O**utput value of analog input value ***C1***.
9+5×C1	OERR*C1*	OB	**ERR**or bit for binary **O**utput value ***C1***.

Initialization

All outputs and the local memory will be set to “0”.

Function

For special applications a digital input module with supervised and tested signals is needed. Because an appropriate module may not be available, the AX646 module can be used instead. For this purpose the 4…20 mA range needs to be converted as follows:

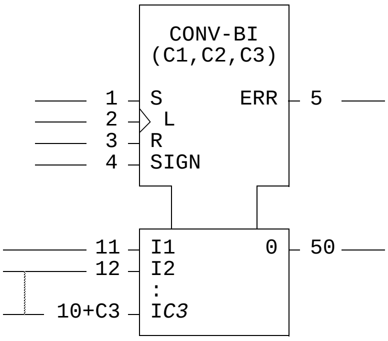
4…8 mA → “0”
12…20 mA → “1”

The CONV-AX element converts the analog signals of AX646 that are supervised and verified by the PC_AX646 PC element to digital values “0” and “1”. CONV-AX can convert up to 16 analog signals to their digital equivalent. CONV-AX loops over all analog input signals, reads the values ACT*n*, I*n*, and IERR*n*, then writes O*n* and OERR*n* (*n* = number of signal).

Code Converter, Boolean to Integer CONV-BI

Summary

CONV-BI (**CONV**erter **- B**oolean to **I**nteger) converts data from BC, BCD, 1-of-N, Gray code or 16/32 Bit Packing, formed from Boolean variables into integers. The element has a memory function for storage of converted data.



PC element CONV-BI

Call CONV-BI (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type of output	I, IL
C2	Type of conversion 1 for BC 2 for BCD 3 for 1-of-N code 4 for Gray code 5 for Bit Packing(1)	1…5
C3	Number of inputs for data to be converted. 1…31 If C2=5 then C3 must be 15 or 31.	1…31

(1) When C2=5 the output O cannot be interpreted as I/IL by the aid.

Number of inputs required with

Code	Number of inputs required with	
	I	IL
BC	15	31
BCD	18	31
1-of-N	31	31
Gray	15	31
Bit packing	15 + SIGN	31 + SIGN

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for storage of a new value each time the element is executed. When this input is reset (to 0) the latest calculated value will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of data.
3	R	IB	**R**eset. Input for clearing of the output. This input overrides the S and L inputs.
4	SIGN	IB	Input set (to 1) with negative data values. If C2 = 5 SIGN is used as the most significant input to get a 16/32 bit conversion.
5	ERR	OB	**ERR**or. Output which is set when more than one input is set with conversion of 1-of-N code or if the integer value to be stored at the output cannot be represented by the data type.
11	I1	IB(1)	**I**nput **1**.
12	I2	IB(1)	**I**nput **2**.
: 10+C3	I*C3*	IB(1)	**I**nput ***C3***.
50	O	O*C1*	**O**utput. Output for converted data.

(1) The data at each of these inputs is of Boolean type but together they form a value of the type specified by parameter C1.

Function

The element converts binary code (BC), binary coded decimal code (BCD), 1-of-N code, Gray code, or uncoded Booleans (Bit Packing) to integers of data type I (16 bit) or IL (32 bit). The SIGN input can be used as a sign bit (C2 = 1, 2 or 3) or as most significant bit (C2 = 5).

Examples of Input Values with Different Decodings

Input values with different decodings

Inputs	Input integer value			
	BC with sign	BC without sign	BCD	1-of-N
I1	1	1	1	1
I2	2	2	2	2
I3	4	4	4	3
I4	8	8	8	4
I5	16	16	10	5
I6	32	32	20	6
I7	64	64	40	7
I8	128	128	80	8

Input values with different decodings (Continued)

Inputs	Input integer value			
	BC with sign	**BC without sign**	**BCD**	**1-of-N**
I9	256	256	100	9
I10	512	512	200	10
: I31	: 2^{30}	: 2^{30}		
SIGN	sign	2^{31}		

Only one of the inputs may be set with 1-of-N code. All inputs have a fixed value with BC, BCD and 1-of-N codes but not with Gray code.

BC, I				Gray code, I				1-of-N-code, I														
4	**3**	**2**	**2**	**4**	**3**	**2**	**1**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	1	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	1	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Storage of Data

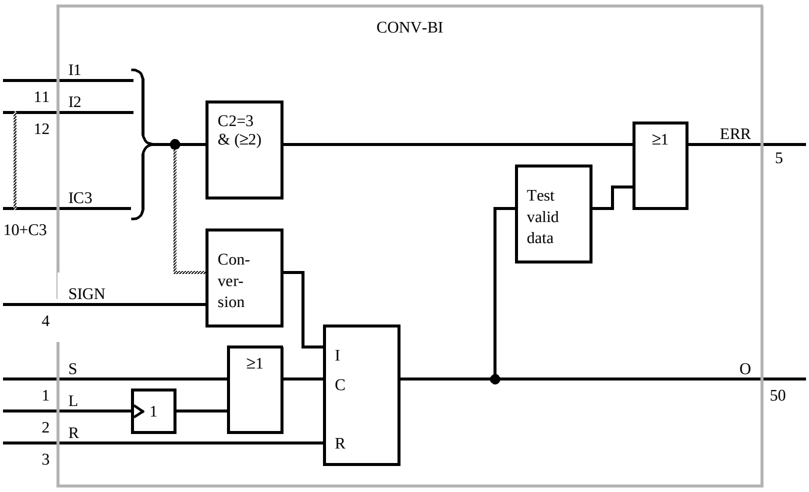
The code at the inputs I1…I*C3* is converted and stored immediately the input L is set. If input S is set, the input code is converted and the integer is stored each time the element is executed. When S is reset (to 0) after having been set, the data stored most recently remains. The input S overrides the input L, that is when S is set, L has no effect.

Clearing

The input R clears the output and prevents all further storage of data while R is set.

Supervision

When converting 1-of-N code the setting of only one of the inputs 1…I*C3* is supervised. If two or more inputs are set, the value of the input signal with the lowest number is stored. The error signal output ERR is also set. If the integer value to be stored at the output exceeds the value which can be represented, the error signal output is set. In addition, the output is limited to the upper or lower limit value.

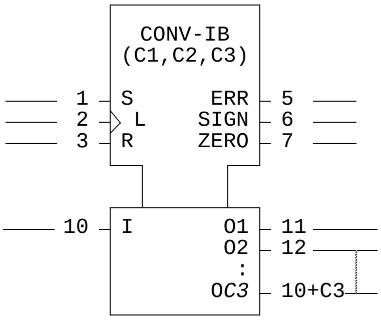


Function diagram

Code Converter, Integer to Boolean CONV-IB

Summary

CONV-BI (**CONV**erter **- I**nteger to **B**oolean) converts data from integer into BC, BCD, 1-of-N, Gray code 16/32 bit BC formed from Boolean variables. The element has a memory function for storage of converted data.



PC element CONV-IB

Call CONV-IB (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type	I, IL
C2	Type of conversion 1 for BC 2 for BCD 3 for 1-of-N code 4 for bit unpacking(1)	1…4
C3	Number of converted data.	1…32

(1) When C2=4 the input 1 cannot be interpreted as I/IL by the tool.

Number of outputs required with different codes

Code	Number of outputs required with	
	I	IL
BC	15	31
BCD	18	32
1-of-N	31	32
Bit packing	15 + SIGN	31 + SIGN

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for storage of a new value each time the element is executed. When this input is reset (to 0) the latest calculated value will remain at the outputs.

Terminal Description (Continued)

No	Name	Type	Description
2	L	IB	**L**oad. Dynamic input for loading of data.
3	R	IB	**R**eset. Input for clearing of the outputs. This input overrides the S and L inputs.
5	ERR	OB	**ERR**or. Output which is set when the integer value to be converted exceeds the value which can be represented at the outputs.
6	SIGN	OB	Output set to 1 with negative data values. If C2 = 4 SIGN is interpreted as bit 16/32.
7	ZERO	OB	Output set to 1 when the value at the input I is zero.
10	I	IC1	**I**nput. Input for the integer value to be converted.
11	O1	OB(1)	**O**utput **1**.
12	O2	OB(1)	**O**utput **2**.
: 10+C3	O*C3*	OB(1)	**O**utput ***C3***.

(1) The data at each of these outputs is of Boolean type but together they form a value of the type specified by parameter C2.

Function

The element converts integer values of the data types I or IL into Boolean variables using binary code (BC), binary coded decimal code (BCD), 1-of-N code or uncoded unpacking. The code conversion to be performed is specified by the call parameters C1 and C2. With positive data values the output SIGN is reset (to 0) and with negative values, set (to 1). For bit unpacking SIGN is used as the most significant bit.

Examples of Output Values with Different Decodings

Output	Output integer value		
	BC	BCD	1-of-N-code
O1	1	1	1
O2	2	2	2
O3	4	3	3
O4	8	4	4
O5	16	5	5
O6	32	6	6
O7	64	7	7
O8	128	8	8
O9	256	9	9
O10	512	10	10

Examples of Output Values with Different Decodings

Output	**Output integer value**		
	BC	**BCD**	**1-of-N-code**
: SIGN(1)	: 2^{31}		

(1) SIGN is used as a sign for BC, BCD and 1-of-N.Only one of the outputs is set with 1-of-N code.

Storage of Data

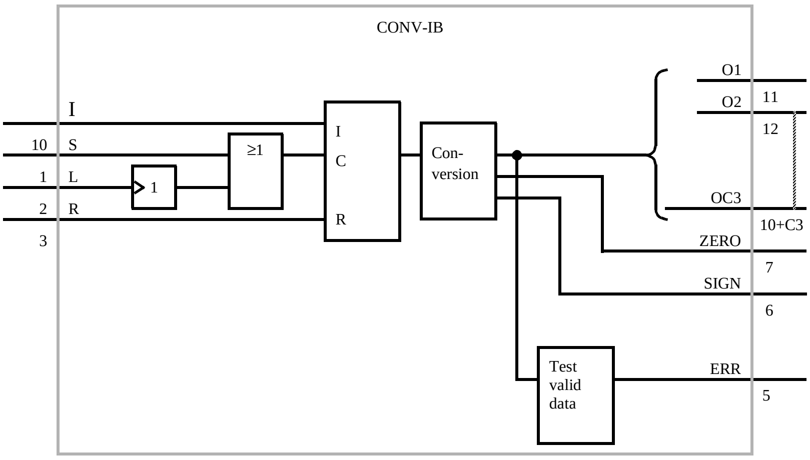
The integer at the input I is converted immediately the input L is set. If input S is set, the input code is stored and the result is converted each time the element is executed. When S is reset (to 0) after having been set, the data stored most recently remains until the element is once more executed with one of the inputs S, L or R set. The input S overrides the input L, that is when S is set, L has no effect.

Clearing

The input R clears the output and prevents all further storage of data while R is set.

Supervision

When converting data from integers to 1-of-N code, the numbers are restricted to -32 to +32. If the integer value to be converted exceeds the value which can be represented at the outputs, the error signal output is set. In addition, the output is limited to the upper or lower limit value.

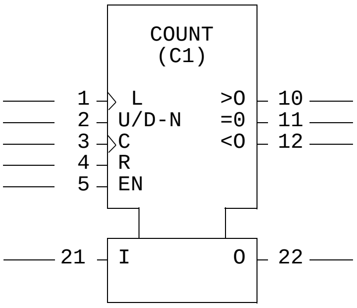


Function diagram

Counter COUNT

Summary

COUNT (**COUNT**er) is a presettable counter for counting pulses, up or down. The counter also monitors the relation of the counter value to 0.



PC element

Call COUNT (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL

Terminal Description

No	Name	Type	Description
1	L	IB	**L**oad. **L**oads the counter with the value at input I.
2	U/D- N	IB	**U**p/**D**own-**N**. Input which determines if the counter is to count up (U/D-N=1) or down (U/D-N=0).
3	C	IB	**C**lock. When this input changes from 0 to 1 the counter counts up or down in accordance with the status of the input U/D-N.
4	R	IB	**R**eset. Input which clears the counter and prevents all further counting or loading.
5	EN	IB	**EN**able. Input which is to be 1 to permit counting or loading. Reset execution is, however, performed independently of EN.
10	>0	OB	Output which is set when the value of the counter is greater than 0.
11	=0	OB	Output which is set when the value of the counter is 0.
12	<0	OB	Output which is set when the value of the counter is less than 0.
21	I	I*C1*	**I**nput. Input for new value when loading.
22	O	*OC1*	**O**utput. Output for counter value.

Function

The counter value increases or decreases by 1 immediately the input C is set. The value increases if U/D–N=1 and decreases if U/D–N=0. The duration of the counter period may not be less than 2 × the cycle time of the program.

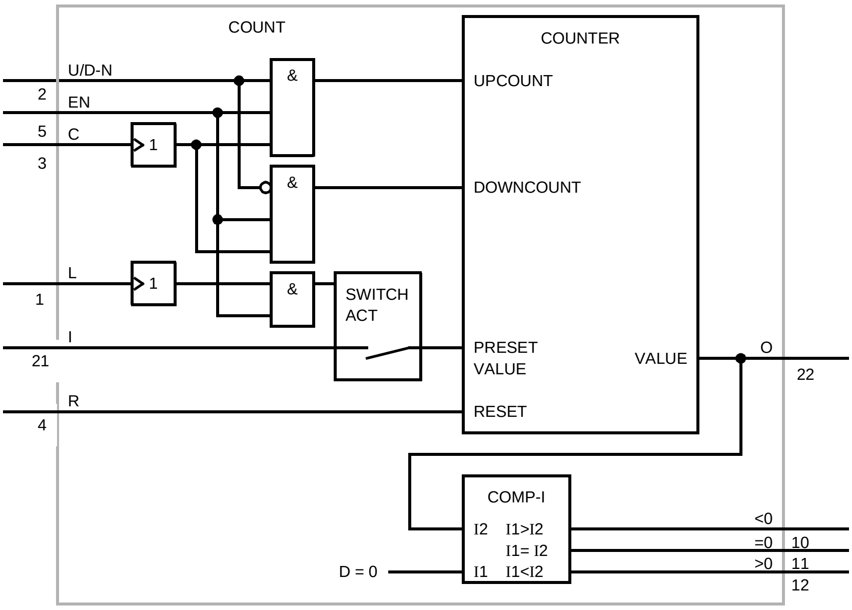
When the input L is set, the counter is loaded with the value at input I. If both input L and input C are set simultaneously, the counter is first loaded after which an up or down count is performed. The input EN must be set for the counter to count or load a new value

Clearing

The input R clears the counter and prevents all further counting or loading. R overrides EN.

Supervision

The status outputs specify the relation of the counter value to zero (>0, =0, <0). When the counter reaches its least or greatest value for the data type, all counting ceases.

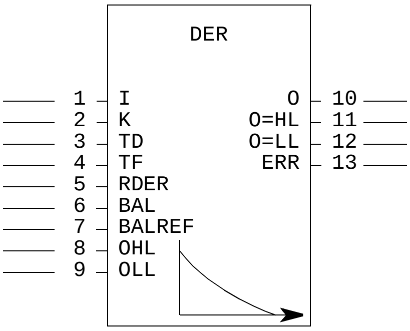


Function diagram

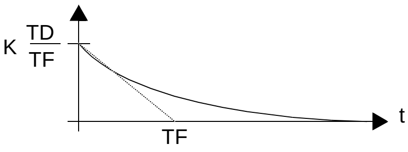
Derivator DER

Summary

DER (**DER**ivator) is used to give derivation effect.
The derivation effect can be limited with the filter function, which serves as a low pass filter. The output signal can be limited with limit values specified at special inputs.
The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from static states are bumpless.



PC element DER



Step response

Call DER

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	TD	ITR	**T**ime **D**erivation. Input for time constant for derivation.
4	TF	ITR	**T**ime **F**ilter. Input for filter time constant.
5	RDER	IB	**R**eset **DER**ivator. Input for clearing derivator.
6	BAL	IB	**BAL**ance. Input for activation of following.
7	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
8	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
9	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output is limited to the upper limit value.
12	O=LL	OB	**O**utput = **L**ow **L**imit. Output which is set to 1 if the output is limited to the lower limit value.

Terminal Description

No	Name	Type	Description
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.

Function

The step response in the time plane for a derivator is:

$O(t) = K\times(TD/TF)\times e^{-t}/TF \times I(t)$ where $I(t)$ specifies the magnitude of the step.

The transfer function for a DER function is:

$G(s) = K\times(s \times TD)/(1+s \times TF)$

This has been implemented in the DER element as a recursive algorithm. The design of the algorithm is such that normal functioning is maintained even during limiting. This ensures a controlled return to a dynamic state.

Gain, Derivation Time, Filter Time and Sampling Time

Certain constants are precalculated to make the execution time of the element as short as possible. The results are stored internally in the element. These constants are recalculated if TD, TF or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating a test is performed to check whether TD and $TF \geq 2 \times TS$. If this is not the case TD and/or TF is set equal to $2 \times TS$.

Clearing of the Derivator

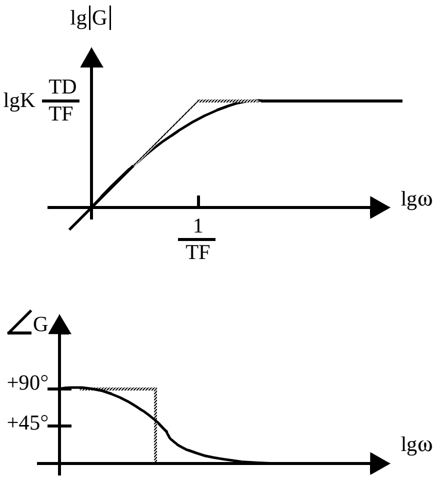
Both the output O(t) and the internal state of the output are cleared when RDER goes to 1.

Following

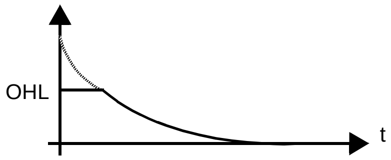
If BAL is set to 1, the derivator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. Return to dynamic state is bumpless.

Limitation Function

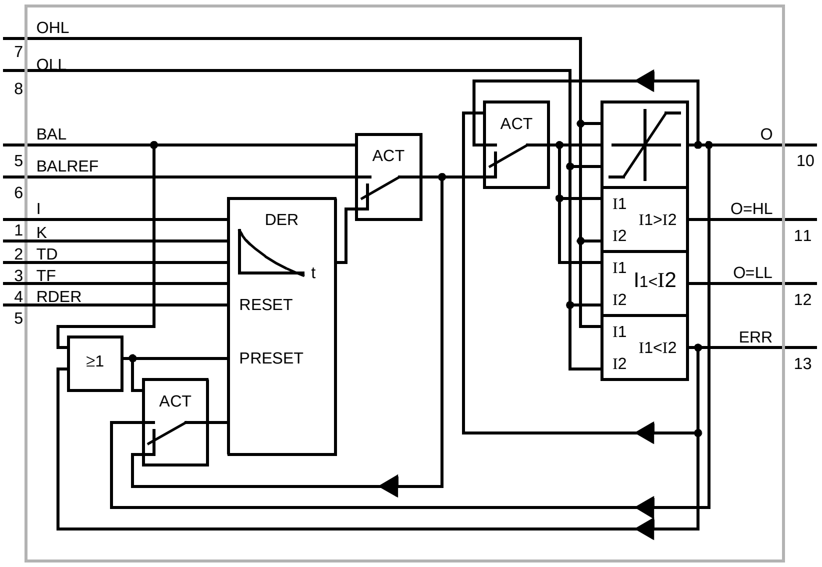
The limitation function limits the output signal to the values at the inputs OHL for upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following above.



Bode diagram



Function in a limiting state

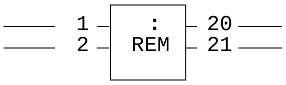


Function diagram

Divider DIV

Summary

DIV is used for division of two integers or real numbers. When dividing integers, the quotient is obtained with the remainder at a separate input.



PC element DIV

Call DIV (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	–	I*C1*	Input for dividend.
2	–	I*C1*	Input for divisor.
20	–	O*C1*	Output for quotient.
21	REM	O*C1*	**REM**ainder. Output for remainder. Applies only with division of integers.

Function

The value at input 1 is divided by the value at input 2. The quotient is stored at output 20. When dividing integers, the remainder is stored at the output REM.

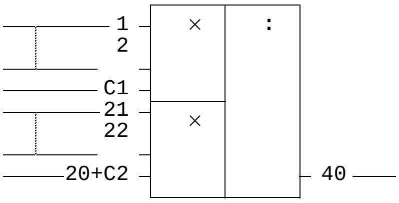
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Divider with Multiple Inputs DIV-MR

Summary

DIV-MR (**DIV**ider **-** with **M**ultipliers **R**eal numbers) is used for division of two products of real numbers.



PC element DIV-MR

Call DIV-MR (C1, C2)

Parameter	Significance	Permissible values
C1	Number of inputs in the first multiplication element.	1…19
C2	Number of inputs in the second multiplication element.	1…19

Terminal Description

No	Name	Type	Description
1	–	IR	Factor for dividend.
2	–	IR	Factor for dividend.
: C1	–	IR	Factor for dividend.
21	–	IR	Factor for divisor.
22	–	IR	Factor for divisor.
: 20+C2	–	IR	Factor for divisor.
40	–	OR	Output for quotient.

Function

The product of the real values at the inputs 1…*C1* is divided by the product of the real values at the inputs 21…20+C2. The quotient is stored at the output 40.

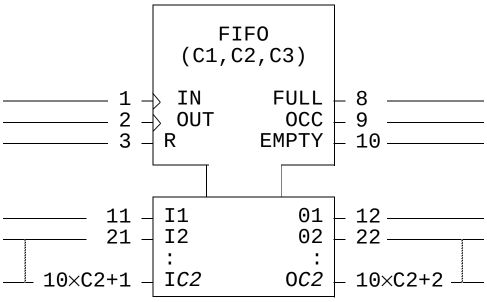
Overflow

If the maximum positive or negative real values are exceeded, the output is limited to the highest or lowest representable value respectively.

Queue Register FIFO

Summary

The queue register FIFO (**F**irst **I**n **F**irst **O**ut) can be used for storage of data to be used later and irrespective of when the data was stored. The different queues in the element work in parallel, that is reading in, reading out, and so on. are performed at the same time in all of the queues. Data may be of the types integer, real number, Boolean or time.
A maximum of 9 queues each with a maximum of 64 queue places can be selected.



PC element FIFO

Call FIFO (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of queues	1..9
C3	Number of queue places in each queue	2…64

Terminal Description

No	Name	Type	Description
1	IN	IB	Dynamic input for entry of data in the queues. Data is read from the inputs I1…I*C2*.
2	OUT	IB	Dynamic input for reading data from the queues.
3	R	IB	**R**eset. Input which clears the queue register and prevents all further entry.
8	FULL	OB	Output which is set when the queue register is full.
9	OCC	OI	**OCC**upied. Output which specifies how many queue places are occupied.
10	EMPTY	OB	Output which is set when the queue register is empty.
11	I1	I*C1*	**I**nput **1**. Input data to queue no. 1.
12	O1	*OC1*	**O**utput **1**. Output data from queue no. 1.
21	I2	I*C1*	**I**nput **2**. Input data to queue no. 2.
22	O2	*OC1*	**O**utput **2**. Output data from queue no. 2.
: 1+10×C2	I*C2*	I*C1*	**I**nput ***C2***. Input data to queue no. C2.
2+10×C2	*OC2*	*OC1*	**O**utput ***C2***. Output data from queue no. C2.

Function

Each queue has a data input I. It is used when data is to be entered at the end of the queue. Each queue has a data output O. Data from place 1 in each queue is continuously available at the outputs O1…O*C2*. If both of the dynamic inputs (IN and OUT) are set during the same program cycle, entry is performed first and then reading out.

Entry of Data

Data at the I inputs are placed last in the queues when input IN is set. When the queue register is empty, data is stored at queue place 1, then at queue place 2, and so on.

Output of Data

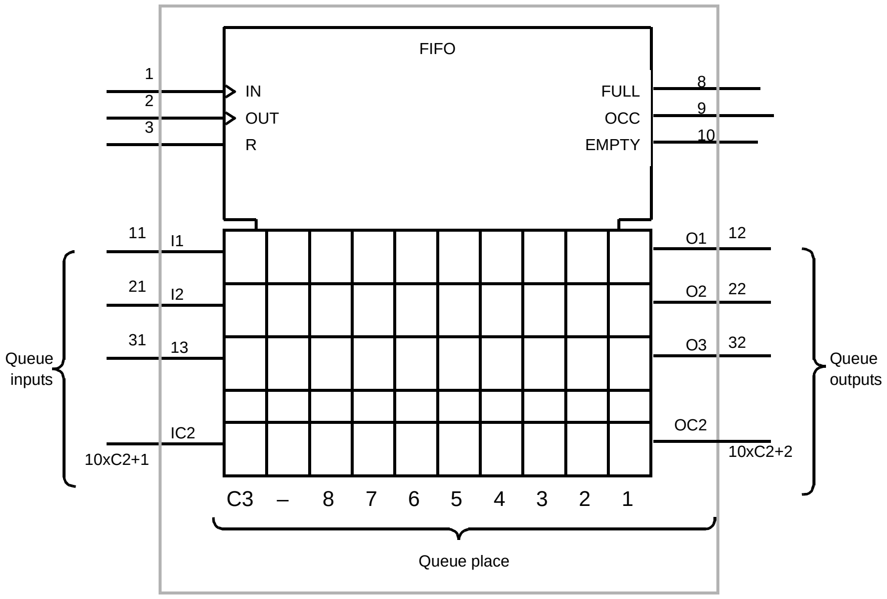
When input OUT goes to 1, data at each used place in all the queues is moved to the place with the number immediately below. 0 is written at the last place used. Data which was previously stored at queue place 2 is now at queue place 1 and can be read at the outputs O1…O*C2*.

Clearing of the Complete Register

When the input R is set, the complete contents of the queues are cleared. All further entry is blocked.

Supervision

How many queue places are occupied is indicated continuously by the output OCC. When the queue register is full, the output FULL is set and when the queue register is empty, the output EMPTY is set.

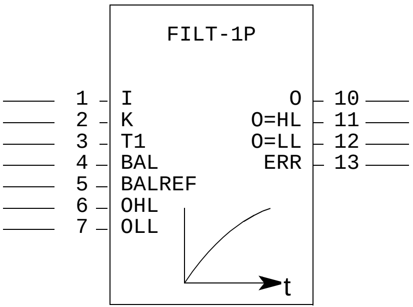


Function diagram

Filter FILT-1P

Summary

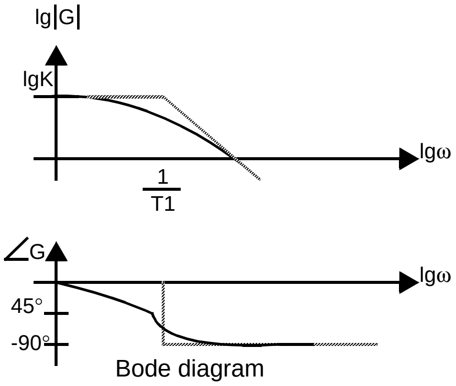
FILT-1P (**FILT**er **- 1 P**ole) is used as a single pole low pass filter. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from static states are bumpless.



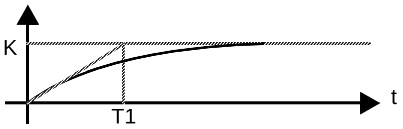
PC element FILT-1P

Call FILT-1P

Diagrams



Bode diagram



Step response

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	T1	ITR	**T**ime **1**. Input for filter time constant.
4	BAL	IB	**BAL**ance. Input for activation of following.
5	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
6	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
7	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.

Terminal Description (Continued)

No	Name	Type	Description
12	O=LL	OB	**O**utput = **L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.

Function

The step response in the time plane for a single pole low pass filter is: $O(t) = I(t)\ K\ (1 - e^{-t/T1})$. The transfer function for a single pole low pass filter is: $G(s) = K(1/(1+s\times T1))$. This has been implemented in the FILT-1P element as a recursive algorithm.

Gain, Filter Time and Sampling Time

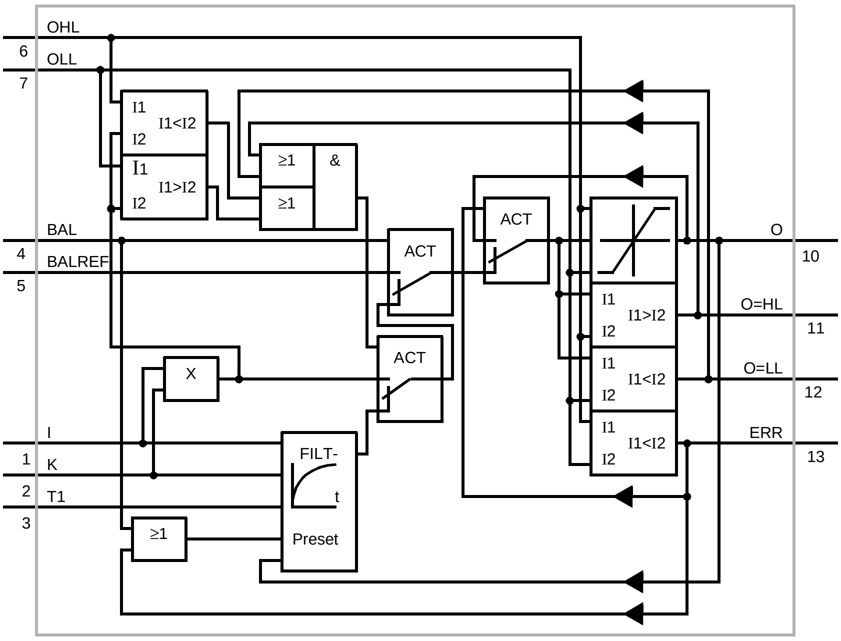
Certain constants are precalculated to make the execution time of the element as short as possible. The result is stored internally in the element. These constants are recalculated if T1 or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating, a test is performed to check if $T1 > 2 \times TS$. If this is not the case T1 is set equal to $2 \times TS$.

Following

If BAL is set to 1, the filter immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. Return to dynamic state is bumpless.

Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit value and OLL for the lower limit value. If the actual value exceeds the upper limit value, the output O=HL is set to 1 and if it falls below the lower limit value, the output O=LL is set to 1. When the limitation status has been detected, a check is made each time the element is executed to determine whether $K \times I(t)$ exceeds the output signal limitations. If such is the case, the limitation status remains. If not, the calculation of the output signal is performed by the algorithm in the normal way. Return from limitation to a dynamic state is bumpless. The element checks that the upper limit value OHL is greater that the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following above.



Function diagram

Function Header FUNCM

Summary

FUNCM (**FUNC**tion **M**odule) is used for structuring a PC program.

PC element FUNCM

Call FUNCM

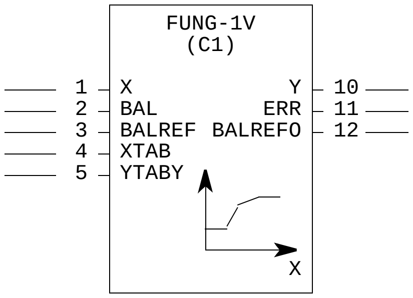
Function

The purpose of FUNCM is to control the arrangement of the documentation. The element cannot affect the execution.
A function module can be used to provide a subdivision of a PC program, a control module, a slave or a sequence step in several functionally associated parts. A function module can itself contain a control module, slave sequence or a new function module.

Function Generator FUNG-1V

Summary

FUNG-1V (**FUN**ction **G**enerator - **1** **V**ariable) is used for generation of an optional function of one variable, y = f(x). The function is described by a number of co-ordinates. Linear interpolation is used for values between these coordinates. A maximum of 255 coordinates can be specified.



PC element FUNG-1V

Call FUNG-1V (C1)

Parameter	Significance	Permissible values
C1	Number of coordinates which describe the function.	2…255

Terminal Description

No	Name	Type	Description
1	X	IR	**X**-value. Input for X-value.
2	BAL	IB	**BAL**ance. Input for activation of balancing.
3	BALREF	IR	**BAL**ance **REF**erence. Value which the Y-output is to adopt with balancing.
4	XTAB	IGC1R	**X TAB**le. Group data for the X-table with C1 values.
5	YTAB	IGC1R	**Y TAB**le. Group data for the Y-table with C1 values.
10	Y	OR	**Y**-value. Output for Y-value.
11	ERR	OB	**ERR**or. Error signal which is set (to 1) if X is outside the value of XTAB or if Y, on balancing, is outside the YTAB values.
12	BALREFO	OR	**BAL**ance **REF**erence Output. Output for calculated X-value with balancing.

Function

The function generator FUNG-1V for one variable calculates an output signal Y for a value at the input X. The calculation is performed in accordance with a piece by piece linear function which is determined by the vectors XTAB and YTAB. For each X-value in XTAB, there is a corresponding Y-value in YTAB. The Y-value at the output is calculated by means of linear interpolation between the two X-values in XTAB which are nearest the value at the input X. The values in X-tab must be strictly increasing from low to high serial numbers in the table.

Interpolation

The function generated can be illustrated by the following figure. The interpolation is performed as follows:

$y = y_k+(x–x_k)\ (y_{k+1}–y_k)/(x_{k+1}–x_k)$

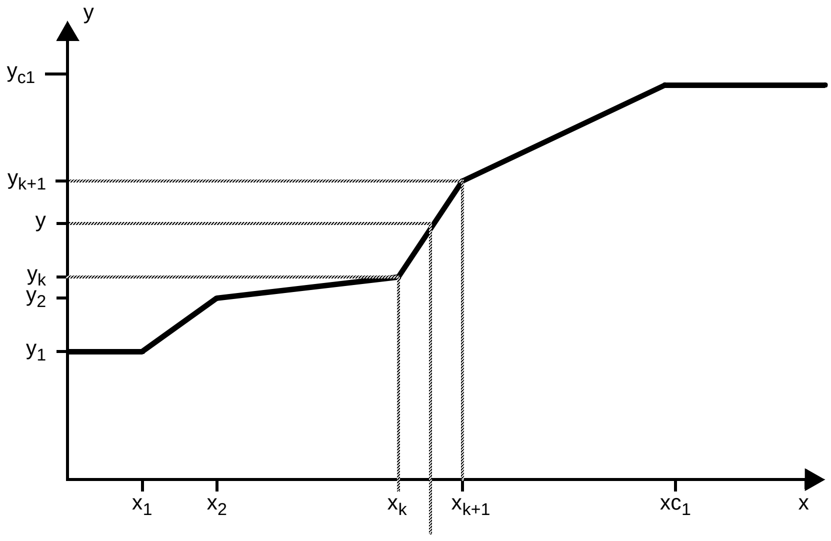
Balancing

On activation of the balancing input BAL, the value at Y is set to the value at the input BALREF. The X-value which corresponds to this Y-value is obtained at the output BALREFO. On balancing, the X-value is calculated by interpolation in the same way as the Y-value is calculated during normal operation. To permit the balancing, the values in YTAB must be strictly increasing or decreasing from low to high serial numbers in the table.

Error Signal

If the input signal X is outside the range defined by XTAB, the ERR output is set to 1. The Y-value is then set to the greatest or lowest value resp. in YTAB.

ERR is also set to 1 if BALREF is outside the YTAB value range when BAL is set to 1. The value at Y is then set to the value at the input BALREF and BALREFO is set to the greatest or lowest value resp. in XTAB.

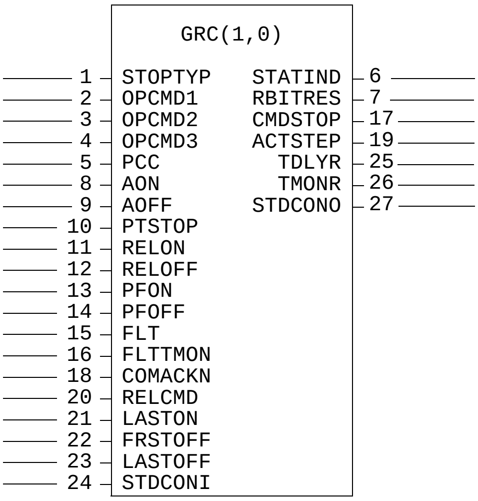


Example of function

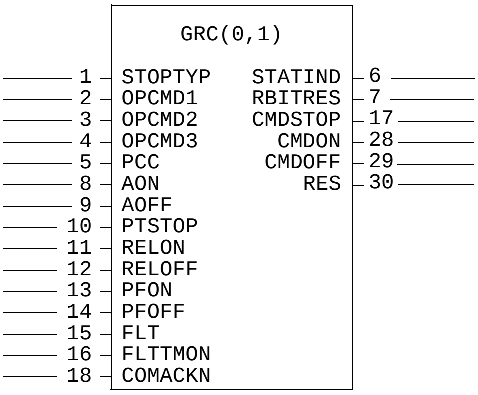
Group Control GRC

Summary

The element GRC (**GR**oup **C**ontrol) is used as central control of a group control. Input and output of commands as well as annunciation functions comply with the requirements of sequential controls and logical controls.



PC element GRC (Sequential Control)



PC element GRC (Logical Control)

Call GRC(C1,C2)

Parameter C1	Parameter C2	Significance
1	0	Sequential control
0	1	Logical control
0	*0*	*Combination not allowed.*
1	*1*	*Combination not allowed.*

The combination of both C1 and C2 define whether the element will be used as sequential control or as logical control.

Terminal Description

No	Name	Type	Description
1	STOPTYP	IB	**STOP TYP**e input. Modes of operation: 1 = STOP, 0 = MANUAL.
2	OPCMD1	IIL	**OP**erator **C**o**M**man**D 1**. Command telegram from MMI.
3	OPCMD2	IIL	**OP**erator **C**o**M**man**D 2**. Command telegram from an additional operator's console.
4	OPCMD3	IIL	**OP**erator **C**o**M**man**D 3**. Command telegram from an additional operator's console with priority.
5	PCC	IB	**P**riority **C**ontrol **C**ondition. Release of OPCMD3 and blocking of OPCMD1 and OPCMD2.
6	STATIND	OIL	**STAT**us **IND**ication output. Status telegram to MMI.
7	RBITRES	OIL	**R**elease **BIT RES**et. Output for resetting the release bit in OPCMD1.
8	AON	IB	**A**utomatic command **ON**.
9	AOFF	IB	**A**utomatic command **OFF**.
10	PTSTOP	IB	**P**ro**T**ection **STOP**. Stopped due to process criteria.
11	RELON	IB	**REL**ease **ON**.
12	RELOFF	IB	**REL**ease **OFF**.
13	PFON	IB	**P**rocess **F**eedback **ON**.
14	PFOFF	IB	**P**rocess **F**eedback **OFF**.
15	FLT	Ib	General **F**au**LT**.
16	FLTTMON	IB	**F**au**LT MON**itoring **T**ime. Monitoring time has expired.
17	CMDSTOP	OB	**C**o**M**man**D** output **STOP**.
18	COMACKN	IB	**COM**mon **ACKN**owledgement of all faults.
If C1=1, the following terminals will be shown:			
19	ACTSTEP	OR	**ACT**ual **STEP** number.
20	RELCMD	IB	**REL**ease for step **C**o**M**man**D**s.
21	LASTON	II	LAST ON. Last step of the running-on program.
22	FRSTOFF	II	**F**i**RST OFF**. First step of the running-off program.
23	LASTOFF	II	LAST OFF. Last step of the running-off program.
24	STDCONI	IG	**ST**andar**D CON**nection **I**n. Connection from the last step.
25	TDLYR	OIL	**D**e**L**a**Y T**ime **R**emaining.
26	TMONR	OIL	**MON**itoring **T**ime **R**emaining.
27	STDCONO	OG	**ST**andar**D CON**nection **O**ut. Connection to the first step.

Terminal Description (Continued)

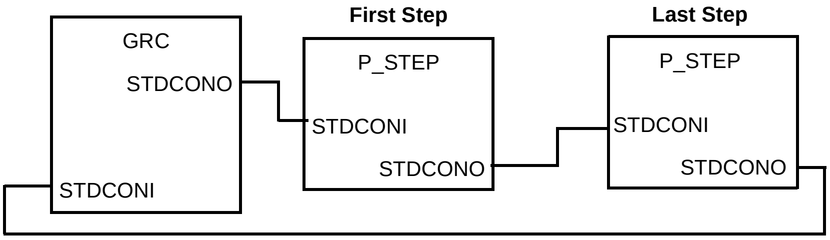
No	Name	Type	Description
If C2 = 1, the following terminals will be shown:			
28	CMDON	OB	**C**o**M**man**D** output **ON**.
29	CMDOFF	OB	**C**o**M**man**D** output **OFF**.
30	RES	OB	**RES**et signal out if neither CMDON nor CMDOFF will be active.

NOTE

Command telegram OPCMD3 can only become effective if a logic 1 signal is present at input PCC. Output STATIND can also be connected to the element LMPD (lamp decoder).

Connections of the Element GRC and P_STEP

Data transfer between the element GRC and the elements P_STEP will be done via the connection STDCONI-STDCONO. All steps and the GRC have to be included in this chain.



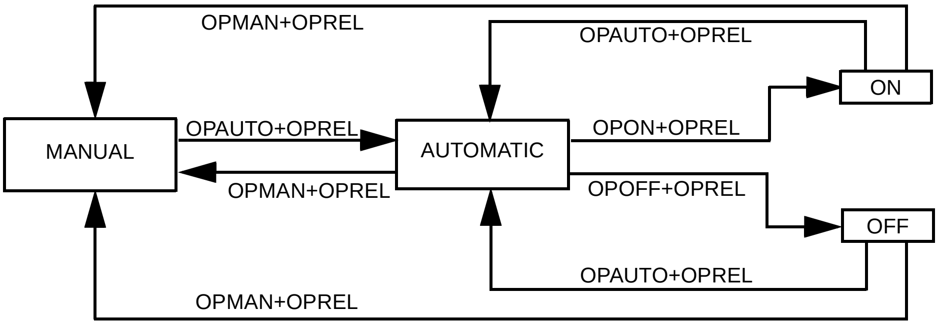
Function

The element can be operated in either one of two operating modes: MANUAL or STOP.

The operating modes will be defined at input STOPTYP. A logic 0 signal at STOPTYP sets the element to MANUAL mode and a logic 1 signal to STOP mode.

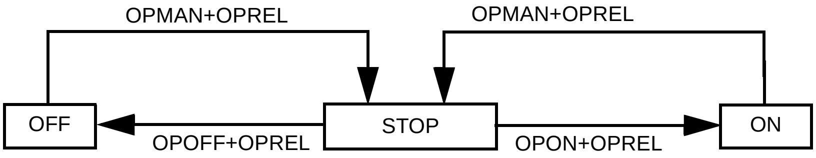
MANUAL mode

Commands OPON and OPOFF can only become effective if release bit OPREL is set and with AUTOMATIC function set before (command OPAUTO and OPREL). Command OPMAN and simultaneously set release bit OPREL reset the function in MANUAL position. The running-on and running-off program will be blocked in MANUAL mode.



STOP mode

Commands OPON and OPOFF can only become effective if release bit OPREL is set. Command OPMAN and simultaneously set release bit OPREL reset the function in STOP position. STOP position is cleared by calling a program.



Input of Commands

Inputs OPCMD1, OPCMD2 and OPCMD3 are used as inputs of the commands.

The commands are given in form of a 32 bit telegram as follows:

Bit	Name	Description
1		
2		
3		
4	OPON	**OP**erator Command **ON**
5	OPOFF	**OP**erator Command **OFF**
6	OPMAN	**OP**erator Command **MAN**ual
7	OPAUTO	**OP**erator Command **AUTO**matic
8	OPACK	**OP**erator **ACK**nowledgement
9		
:	:	:
31		
32	OPREL	**REL**ease for **OP**erator Commands

Except acknowledgement command OPACK, commands can only become effective if corresponding command bit and release bit OPREL are set at the same time.

Super-ordinated Automatic Commands

Super-ordinated automatic commands are given by the commands AUTOMATIC ON:AON and AUTOMATIC OFF:AOFF.

Release Commands

OFF commands OPOFF or AOFF are effective only if the corresponding release RELOFF is set. After the command is stored, the release condition is ineffective.

ON commands OPON or AON are effective only if the corresponding release RELON is set. After the command is stored, the release condition is ineffective.

If there is no need of a release function from the process point of view, inputs RELOFF and RELON have to be connected to a logic 1 signal.

Command Memory

Command memory OFF is set by the input commands OPOFF and OPREL or AOFF in combination with the fulfilled release condition RELOFF.

Command memory ON is set by the input commands OPON and OPREL or AON in combination with the fulfilled release condition RELON.

Both command memories will be reset simultaneously during system start-up and, if a command OPMAN and OPREL is given or a logic 1 signal at the terminal PTSTOP is present. Besides this, they will be reset individually by the corresponding discrepancy signal (refer to item „Discrepancy“) or by the set command memory of the opposite command.

Storage of the Feedback Signals for End Positions

Feedback signals for end positions PFON or PFOFF set the corresponding actual memory ON or actual memory OFF provided that the respective command memory is set.

Both actual memories will be reset during system start-up and, if a command OPMAN and OPREL is given or a logic 1 signal at the terminal PTSTOP is present. They will be reset individually by acknowledgement of the discrepancy signal (refer to item „Acknowledgement“) or by the set command memory of the opposite command.

Discrepancy

When at a set actual memory the feedback signal of the respective end position PFON or PFOFF disappears, discrepancy signal FDY is generated. This discrepancy signal inhibits the corresponding command output CMDON or CMDOFF.

Simultaneously Present Opposing Commands

If two opposing input commands are present simultaneously at OPCMD (OPCMD1:OPON and OPREL, OPCMD2:OPOFF and OPREL) or at the inputs AON and RELON and AOFF and RELOFF, no output command will be put out.

The last and only present input command will be executed.

Priority of Commands

A logic 1 signal at input PCC deactivates command inputs OPCMD1 and OPCMD2 and activates command input OPCMD3.

A logic 0 signal at input PCC deactivates command input OPCMD3. Commands OPCMD1 and OPCMD2 are handled according to the function “first come first serve”.

The acknowledgement command can only be effective at the activated command inputs.

Output of Commands

Output commands CMDON and CMDOFF are carried out if the corresponding command memory is set.

Output command CMDSTOP is carried out if the following conditions are fulfilled:

PTSTOP = 1 and STOPTYP = 1

(OPMAN = 1 and OPREL = 1) and STOPTYP = 0

When output command CMDSTOP is present, output commands CMDON, CMDOFF are inhibited.

Reset command RES is put out if

CMDON = 0 and CMDOFF = 0 and FDY = 0

CMDSTOP = 1.

Output command RBITRES has a logic 0 signal. This output is used to reset a permanently existing telegram at input OPCMD*n* after a cycle has expired. (Usually, RBITRES will be connected to the DB Element which gives data from MMI to input OPCMD*n*.)

Annunciation Signals to Operator Station

Annunciation signals to an Operator Station are given by a 32 bit telegram which is generated at output STATIND.

The content of telegram is as follows:

Bit	Name	Description
1	FFLTSUM	**F**eedback **F**au**LT SUM**mary Alarm
2	FNACK	**F**eedback Summary Alarm **N**ot **ACK**nowledged
3		
4	FFLTT	**F**eedback **F**au**LT T**ime exceeded
5	FDY	**F**eedback **D**iscrepanc**Y**
6	FFLTEL	**F**eedback **F**au**LT EL**ectronic or electric
7		
8		
9		
10	FPFOFF	**F**eedback **P**rocess **F**eedback **OFF**
11	FPFON	**F**eedback **P**rocess **F**eedback **ON**
12	FMAN	**F**eedback **MAN**ual
13	FRES	**F**eedback **RES**et

14	FREQOFF	**F**eedback **REQ**uested **OFF**
15	FREQON	**F**eedback **REQ**uested **ON**
16	FREQMAN	**F**eedback **REQ**uested **MAN**ual
17	FCMDOFF	**F**eedback **C**o**M**man**D** **OFF**
18	FCMDON	**F**eedback **C**o**M**man**D** **ON**
19	FCMDMAN	**F**eedback **C**o**M**man**D** **MAN**ual
20	FAOFF	**F**eedback **A**utomatic Command **OFF**
21	FAON	**F**eedback **A**utomatic Command **ON**
22	FPTSTOP	**F**eedback **P**ro**T**ection **STOP**
23		
24	FRELOFF	**F**eedback **REL**ease **OFF**
25	FRELON	**F**eedback **REL**ease **ON**
26		
27		
28		
29		
30		
31		
32	MMIHLP	**MMI H**e**LP** Bit

Telegram at output STATIND can also be connected to the element LMPD (lamp decoder) when a conventional lamp control shall be used.

Time Monitoring

The active (set) step transmits the monitoring time and the remaining delay time to the element GRC by means of the STDCONI-STDCONO telegram. The element GRC computes the remaining monitoring time and put it out at output TMONR. When the monitoring time has been exceeded (that is the step has not received the proceeding condition within the monitoring time), this will be indicated at bit FFLTT of the STATIND telegram.

The remaining delay time will be put out at output TDLYR.

Indication of a Step

The active (set) step will be indicated at output ACTSTEP.

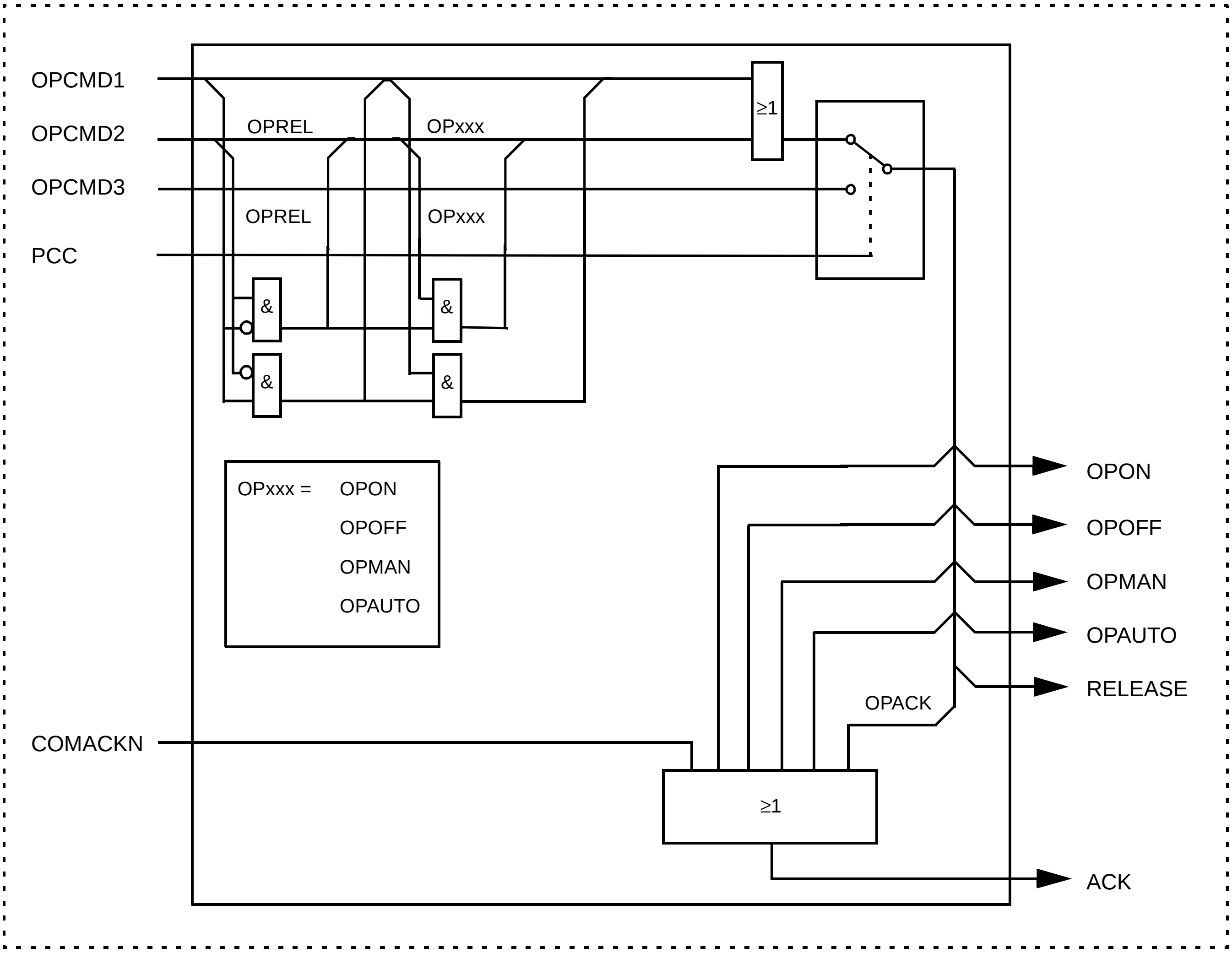
Summary Alarm

Feedback fault signals FFLTT(time exceeded), FDY(discrepancy) and FFLTEL (electronic or electric fault) result in a summary alarm at FFLTSUM.

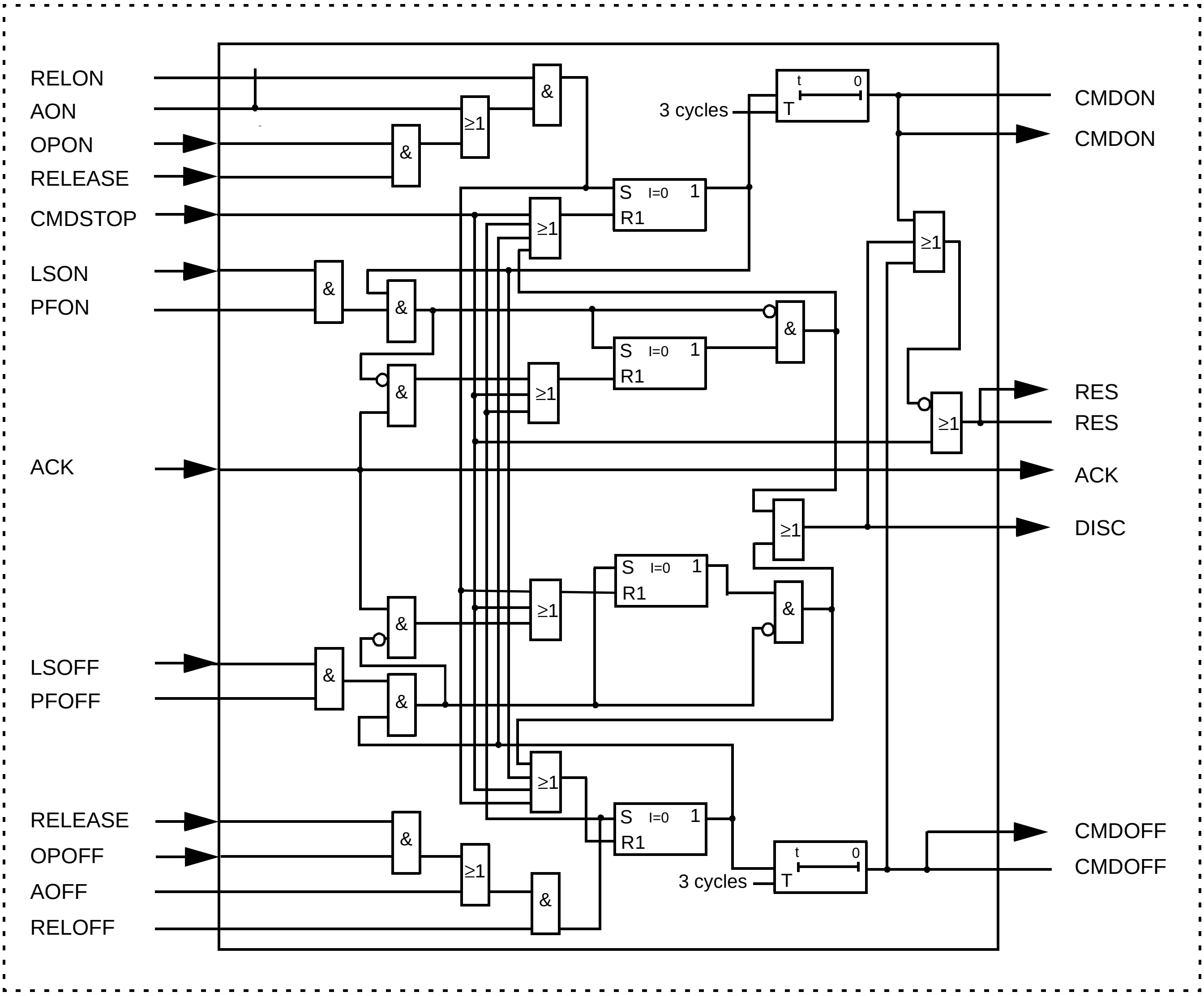
Acknowledgement

Discrepancy signal FDY is acknowledged by commands OPMAN, OPAUTO, OPON, OPOFF, OPACK or acknowledgement input COMACKN. The same is done with acknowledgement of signals FFLTT and FFLTEL.

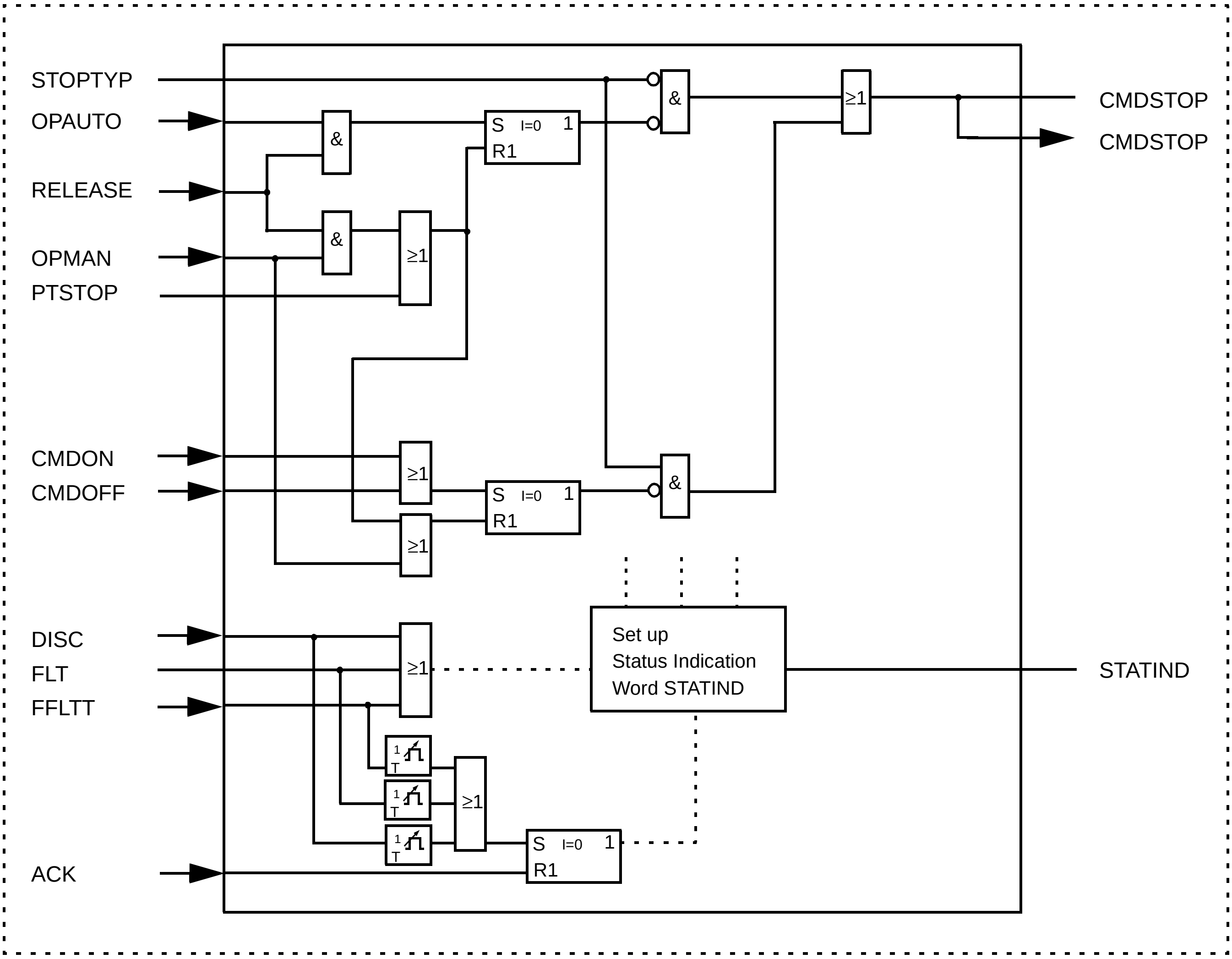
When a not acknowledged fault signal disappears, this state will not be announced.



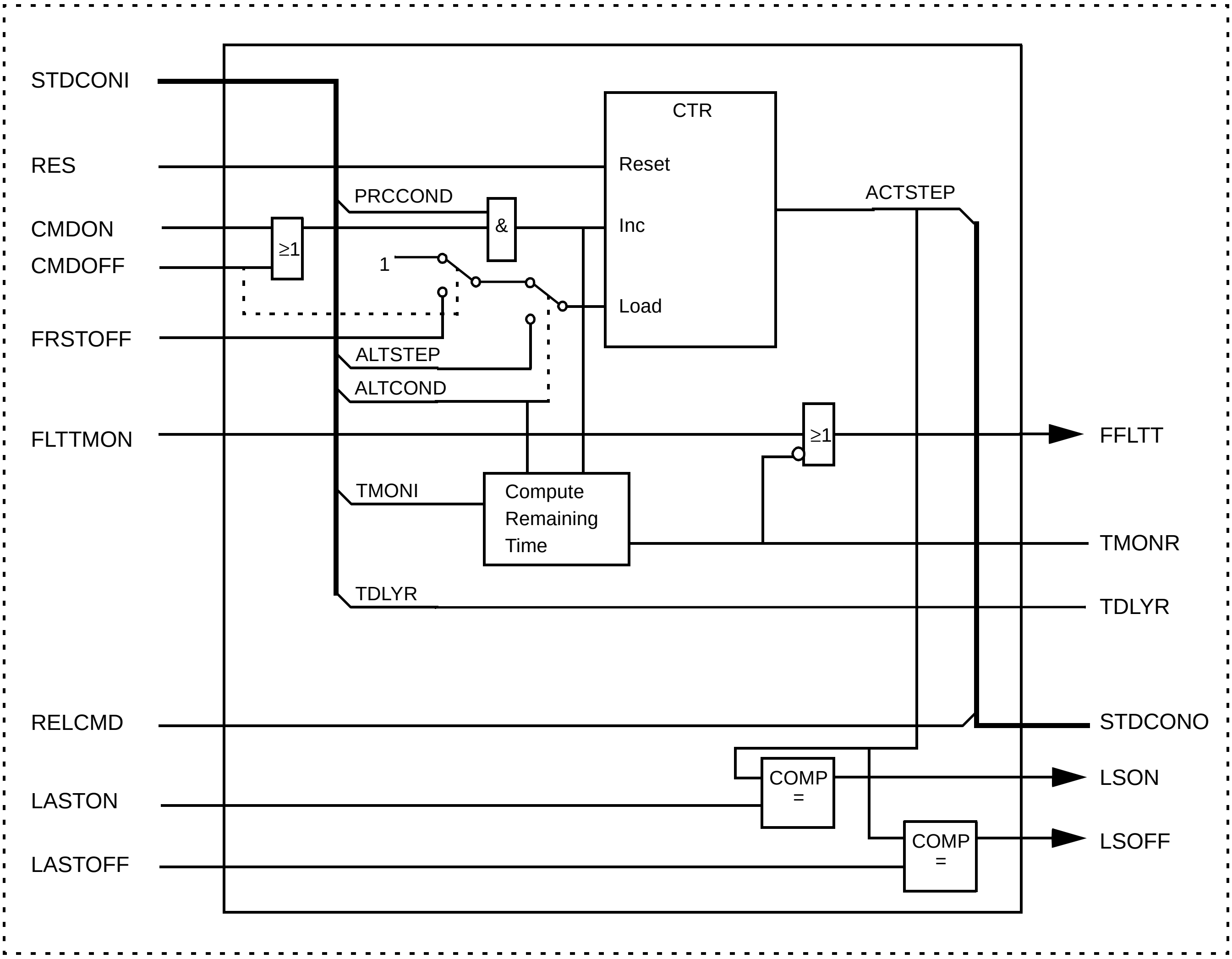
Function diagram: Command logic



Function diagram: Central logic



Function diagram: Central logic (continued)

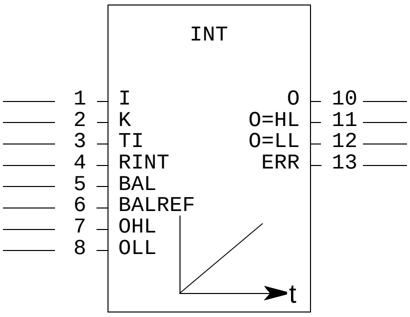


Function diagram: Telegram and time monitoring logic

Integrator INT

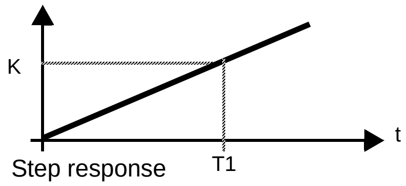
Summary

INT (**INT**egrator) is used to give an integration effect. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function.

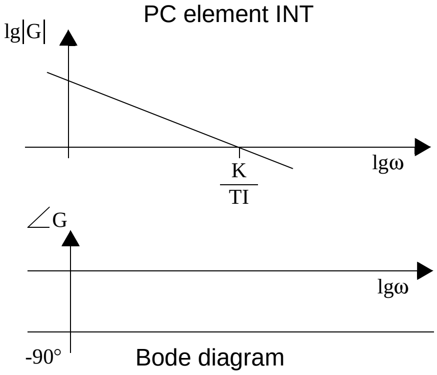


PC element INT

Call INT



Step response



Bode diagram

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input signal.
2	K	IR	Input for setting of gain.
3	TI	ITR	**T**ime **I**ntegration. Input for time constant for integration.
4	RINT	IB	**R**eset **INT**egrator. Input for clearing of integrator.
5	BAL	IB	**BAL**ance REFerence. Input for reference value when following.
6	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
7	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
8	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output is limited to the upper limit value.
12	O=LL	OB	**O**utput = **L**ow **L**imit. Input which is set to 1 if the output is limited to the lower limit value.

Terminal Description

No	Name	Type	Description
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.

Function, Transfer Function

The INT function can be written in the time plane as: $O(t) = K/TI(\int I(\tau)\, d\tau)$. The main property when controlling is that the output signal retains its value when the input signal $I(t) = 0$.The step response in the time plane is $O(t) = k \times I(t) \times t/TI$. The transfer function for an integrator is $G(s) = K(1/s \times TI)$

Gain, Integration Time Constant and Sampling Time

The constant K × TS/TI is precalculated to reduce the execution time of the element to a minimum and the result is stored internally in the element. This constant is recalculated if TI or K is changed by more than 1/128 of their previous values or if the sampling time TS is changed. When recalculating a test is made to see whether $TS/TI < 1$. TS/TI is otherwise set equal to 1.

Clearing of Integrator

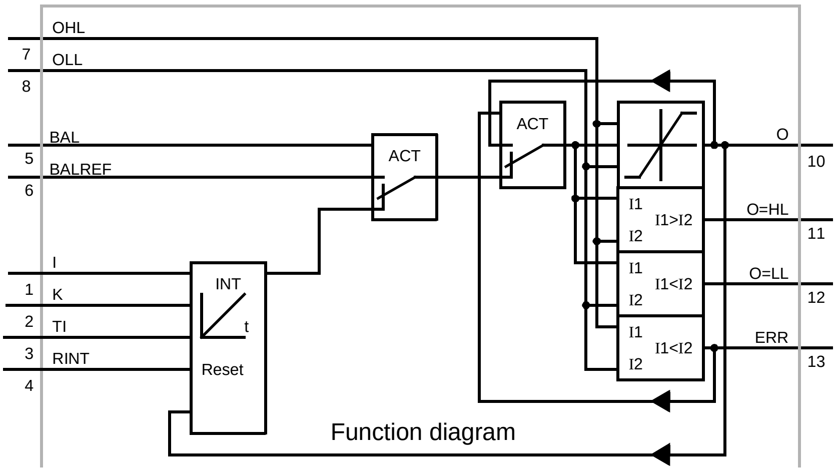
The algorithm is cleared when RINT goes to 1.

Following

If BAL is set to 1, the regulator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. On return to the normal function the value of output O during the last sample in following remains a further sample time, after which integration will be performed for this value.

Limitation Function

The limitation function limits the output signal to the values at the inputs OHL for upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After limitation or error status, normal integration is performed from the current value.



Function diagram

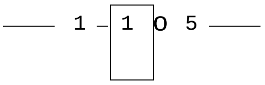
NOTE

The value of output terminal O has only seven significant digits. When integrating small increments the following problem can arise: 1000.0000 + 0.0001 = 1000.0000 and not 1000.0001.

Inverter INV

Summary

INV is used for inverting Boolean variables and particularly when reading signals to the data base.



PC element INV

Call INV

Terminal Description

No	Name	Type	Description
1	–	IB	Input.
5	–	OB	Output of inverted input value.

Function

The output signal from the INV element is set (to 1) if the input signal to the element is 0 and is reset (to 0) when the input signal is 1. See table below.

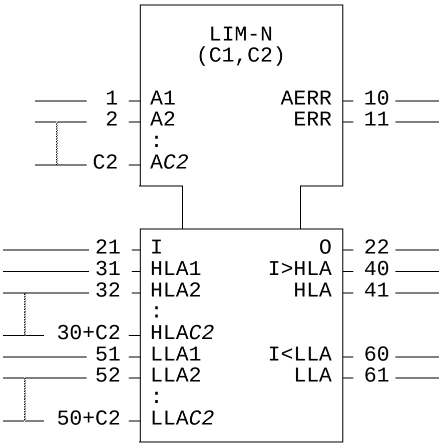
Input and Output Channels

1	5
1	0
0	1

Limiter LIM-N

Summary

LIM-N (**LIM**iter **-** 1-of-**N** address) is used for limitation of integers, real numbers or time values. Several limit values can be selected.



PC element LIM-N

Call LIM-N (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, T, TR
C2	Number of optional limit values	1…9

Terminal Description

No	Name	Type	Description
1	A1	IB	**A**ddress **1**. Input which, when set, limits the output O to the limit values connected to the inputs HLA1 and HLL1.
2	A2	IB	**A**ddress **2**. Input which, when set, limits the output O to the limit values connected to the inputs HLA2 and HLL2.
: C2	A*C2*	IB	**A**ddress ***C2***. Input which, when set, limits the output O to the limit values connected to the inputs HLA*C2* and LLA*C2*.
10	AERR	OB	**A**ddress **ERR**or. Output which is set when 2 or more of the inputs A1…A*C2* are set.
11	ERR	OB	**ERR**or. Output which is set when the limit for high level is less than the limit for low level.
21	I	I*C1*	**I**nput. Input to which the signal to be limited is connected.
22	O	O*C1*	**O**utput. Output for the limited signal.

Terminal Description (Continued)

No	Name	Type	Description
31	HLA1	I*C1*	**H**igh **L**imit **A**ddress **1**. Input for upper limit value which limits the signal when the input A1 is set.
32	HLA2	I*C1*	**H**igh **L**imit **A**ddress **2**. Input for upper limit value which limits the signal when the input A2 is set.
: 30+C2	HLA*C2*	I*C1*	**H**igh **L**imit **A**ddress ***C2***. Input for upper limit value which limits the signal when the input AC2 is set.
40	I>HLA	OB	**I**nput>**H**igh **L**imit **A**ddress. Output which is set when the upper limit of the element is reached.
41	HLA	O*C1*	**H**igh **L**imit **A**ddress. Output which specifies the upper limit where limiting begins.
51	LLA1	I*C1*	**L**ow **L**imit **A**ddress **1**. Input for lower limit value which limits the signal when the input A1 is set.
52	LLA2	I*C1*	**L**ow **L**imit **A**ddress **2**. Input for lower limit value which limits the signal when the input A2 is set.
: 50+C2	LLA*C2*	I*C1*	**L**ow **L**imit **A**ddress ***C2***. Input for lower limit value which limits the signal when the input A*C2* is set.
60	I<LLA	OB	**I**nput<**L**ow **L**imit **A**ddress. Output which is set when the lower limit of the element is reached.
61	LLA	O*C1*	**H**igh **L**imit **A**ddress. Output which specifies the lower limit where limiting begins.

Function

LIM-N is used to limit up to 9 different limits. Boolean output signals are given when the output is limited.

Selection of Limit Value

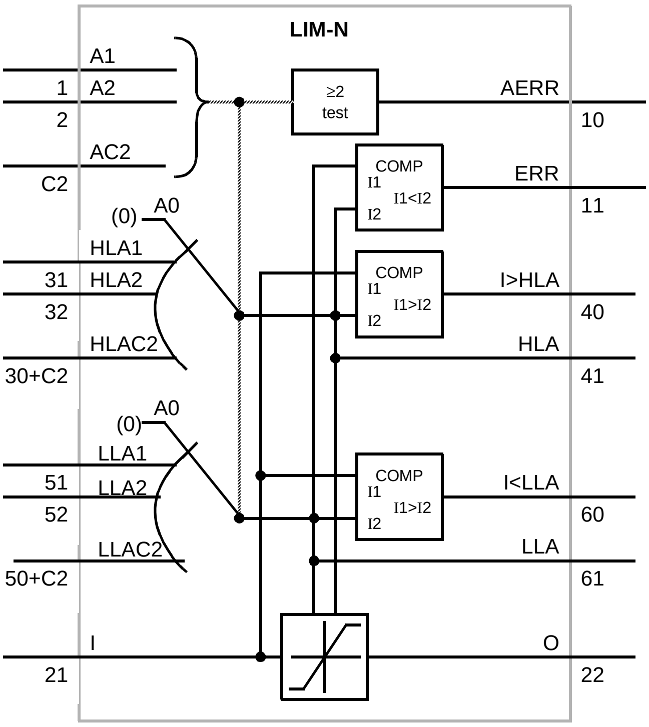
Which of the limit value inputs HLA1…HLA*C2* or LLA1…LLA*C2* is to limit the value at output O is selected with the inputs A1…A*C2*. If the input A1 is 1, the output is limited by HLA1 and LLA1, if A2 is 1, the output is limited by HLA2 and LLA2, and so on. If none of the inputs A1…A*C2* is 1, O is limited to the data value 0. If 2 or more of the inputs A1…A*C2* are set at the same time, the output is limited by the limit values corresponding to the lowest numbered set input. The error signal output AERR is set at the same time.

Limiting

When the input I exceeds the selected limit, the output O is limited to the limit value. One of the outputs I>HLA or I<LLA will then be set depending on which limit was exceeded. The value of the current limits for high and low level where limiting begins can be read at the outputs HLA and LLA.

Supervision of Limit Values

The element checks that the limit value HLA is greater than the limit LLA. If HLA is less than LLA, the error signal output ERR is set. The output O and the limit value outputs I>HLA and I<HLL are retained from the sample before that in which the error status developed.

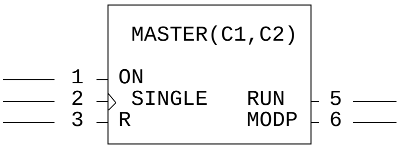


Function

Master Header MASTER

Summary

The master header MASTER is used for the execution control of subordinate slave modules.



PC element MASTER

Call MASTER (C1, C2)

Parameter	Significance	Permissible values
C1	Cyclicity in ms	Allowed values are between 2 ms and 20000 ms in steps of 2 ms
C2	Place in the cycle time table and/or scheduling strategy	1…251: place in cycle time table, cyclic activation 252: activation at initialization after power down. 254: activation at warm start initialization after power down.

Terminal Description

No	Name	Type	Description
1	ON	IB	Control input which is to be set (to 1) with normal execution.
2	SINGLE	IB	If input ON and R are 0 the elements in the slave modules can be executed once by setting SINGLE to 1.
3	R	IB	**R**eset. Input for clearing of the slave modules.
6	MODP	OB	**MOD**ify **P**ermission. Output that indicates if the master and slave modules may be modified from service aids.

Function

MASTER is a superior execution controlling element for a number of slave modules. How often the master and associated slave modules are to be executed is determined with the call parameter C1, and the place in the cycle time table is determined with the call parameter C2. The sequence in which execution units with the same cycle time are to be executed is based on the place in the cycle time table. When and how the slave modules are to be executed is determined by the control inputs ON, SINGLE and R. The master is always executed regardless of the state of these inputs.

Normal Execution

For normal execution the input ON must be set to 1. The master module and associated slave modules are then executed as determined by the call parameters C1 and C2. The order in which the subordinate slave modules are to be executed is determined by their order in the documentation.

If the ON input is reset after execution of the slave modules, the calculated data remains until the next execution of the slave modules. The ON input overrides the SINGLE input, that is if ON is set to 1. SINGLE has no effect. The slave modules are executed successively in three steps:

1. Entry of variables from the data base and from modules with higher priority.
2. Element execution.
3. Reading of variables to the data base and to modules with higher priority.

SINGLE Execution

If the SINGLE input is set when the ON input is reset, the slave modules are executed once as described above. The calculated data remains until the next execution of the master and slave modules.

Clearing

If the input R is set, the slave modules are executed in the reset mode. This means that all outputs of the elements in the slave modules are given initial values, which in most cases are the 0-value of the data concerned. Input R overrides the inputs ON and SINGLE

RUN

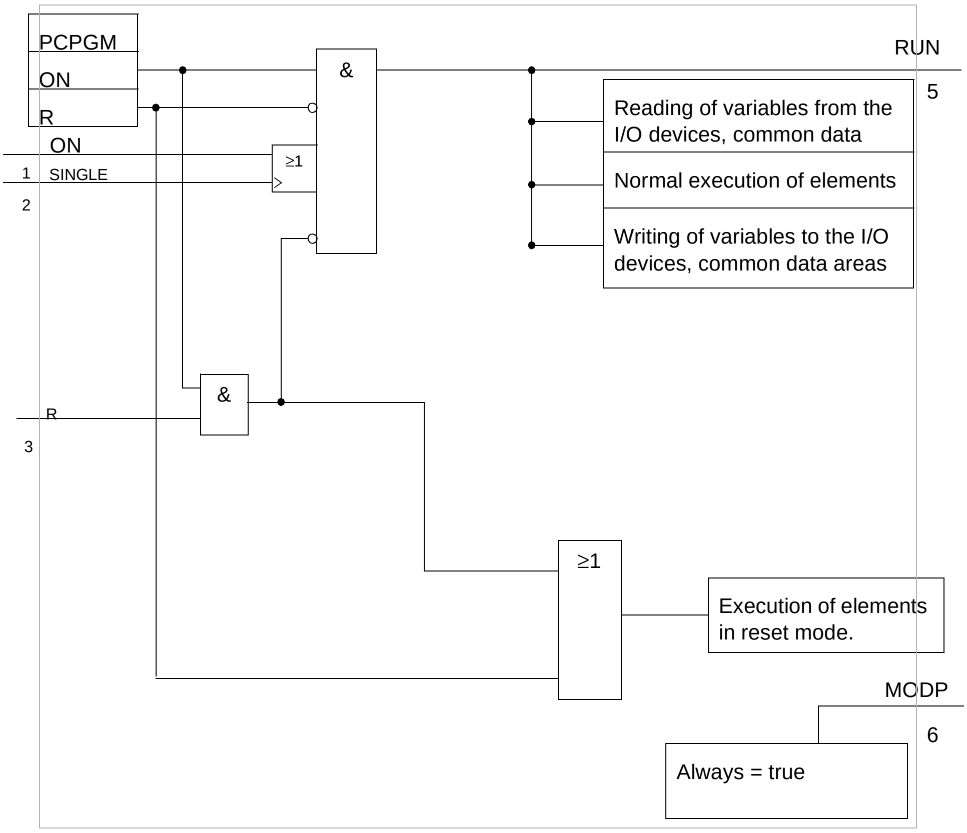
The output RUN is set only during normal execution, that is ON is set or SINGLE execution is performed. With single execution, the RUN output is set for only one cycle.

MODP

Always true.

Effects from PCPGM

The ON and R inputs and the blocking functions on the program header override the inputs on the master module header.
The input ON on PCPGM must be set to permit normal execution of master and slave modules.

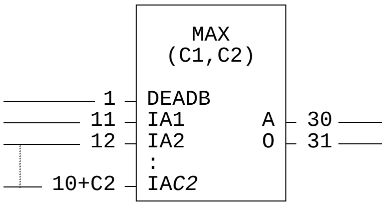

PCPGM
ON
R
ON
1
SINGLE
2
&
≥1
RUN
5
Reading of variables from the I/O devices, common data
Normal execution of elements
Writing of variables to the I/O devices, common data areas
&
R
3
≥1
Execution of elements in reset mode.
MODP
6
Always = true

Function diagram

Maximum Selector MAX

Summary

MAX (**MAX**imum selector) is used to select the largest value of an optional number of integers or real numbers.



PC element MAX

Call MAX (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R
C2	Number of inputs	2…19

Terminal Description

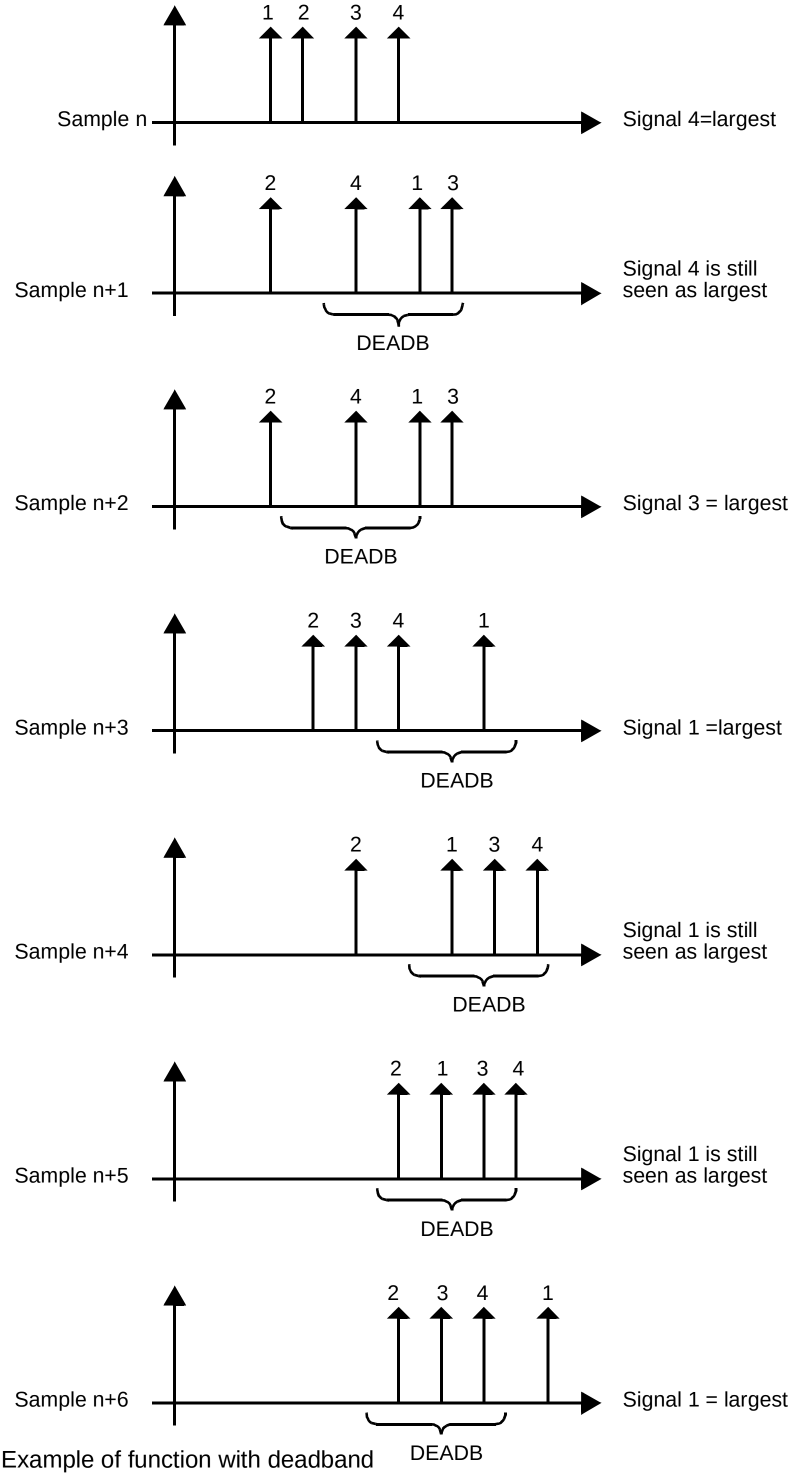
No	Name	Type	Description
1	DEADB	I*C1*	**DEADB**and. Input for deadband.
11	IA1	I*C1*	**I**nput **A**ddress **1**. Input the value of which is compared with the other inputs.
12	IA2	I*C1*	**I**nput **A**ddress **2**. Input the value of which is compared with the other inputs.
: 10+C2	IA*C2*	I*C1*	**I**nput **A**ddress ***C2***. Input the value of which is compared with the other inputs.
30	A	OI	**A**ddress. Output for the number of the input having the greatest value.
31	O	O*C1*	**O**utput. Output for the greatest value.

Function

The values at the inputs IA1…IA*C2* are compared and the greatest value is obtained at the output O. The number of the input with the greatest value is obtained at the output A. If the two largest signal values are equal when the element is executed the first time, the signal with the lowest connection number is selected as the greatest.

Deadband

The deadband specified at the input DEADB is symmetrical around the value of the greatest input. The upper and lower deadband limits are calculated from the value for the largest input in the preceding sample. To prevent rapid changes at the output A, the value at A is retained until the value at the corresponding input is less than the calculated lower deadband limit or until one of the other inputs exceeds the upper deadband limit.

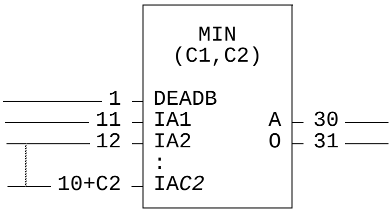


Example of function with deadband

Minimum Selector MIN

Summary

MIN (**MIN**imum selector) is used to select the lowest value of an optional number of integers or real numbers.



PC element MIN

Call MIN (C1, C2)

Parameter	Significance	Permissible values
C1	Data Type	I, IL, R
C2	Number of input	2…19

Terminal Description

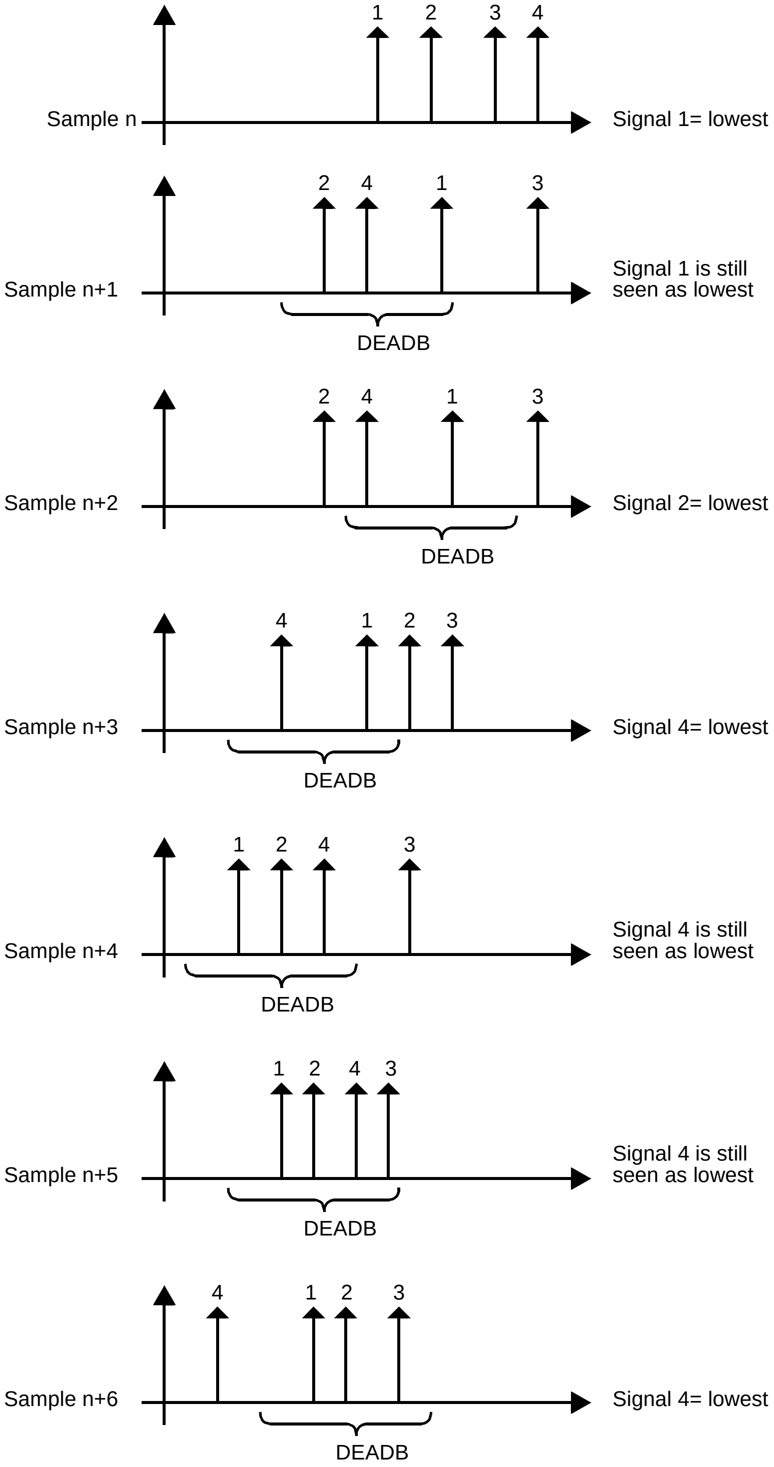
No	Name	Type	Description
1	DEADB	I*C1*	**DEADB**and. Input for deadband.
11	IA1	I*C1*	**I**nput **A**ddress **1**. Input the value of which is compared with the other inputs.
12	IA2	I*C1*	**I**nput **A**ddress **2**. Input the value of which is compared with the other inputs.
: 10+C2	IA*C2*	I*C1*	**I**nput **A**ddress ***C2***. Input the value of which is compared with the other inputs.
30	A	OI	**A**ddress. Output for the number of the input having the lowest value.
31	O	I*C1*	**O**utput. Output for the least value.

Function

The values at the inputs IA1…IA*C2* are compared and the lowest value is obtained at the output O. The number of the input with the lowest value is obtained at the output A. If the two smallest signal values are equal when the element is executed the first time, the signal with the lowest connection number is selected as the smallest.

Deadband

The deadband specified at the input DEADB is symmetrical around the value of the least input. The upper and lower deadband limits are calculated from the value for the lowest input in the preceding sample. To prevent rapid changes at the output A, the value at A is retained until the value at the corresponding input exceeds the calculated upper deadband limit or until one of the other inputs falls below the lower deadband limit.

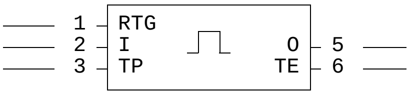


Example of function with deadband

Monostable MONO

Summary

The mono function MONO can be used, for example, for time limiting of operation of outputs and automatic functions. The element can also be used for impulse extension, stall alarm function, and so on.



PC element MONO

Call MONO

Terminal Description

No	Name	Type	Description
1	RTG	IB	**R**e**T**rig**G**. Input for selection of if the MONO function is to be retriggerable or not. If RTG is set, a retriggerable function is obtained.
2	I	IB	**I**nput. Input for start of the MONO function when the input changes from 0 to 1.
3	TP	IT	**T**ime **P**ulse. Input for setting of pulse time.
5	O	OB	**O**utput. Output which is set when the MONO function starts and resets (to 0) when the time set has elapsed.
6	TE	OT	**T**ime **E**lapsed. Output for time which has elapsed when O is set. When the time set (TP) has elapsed TE will stop.

Function

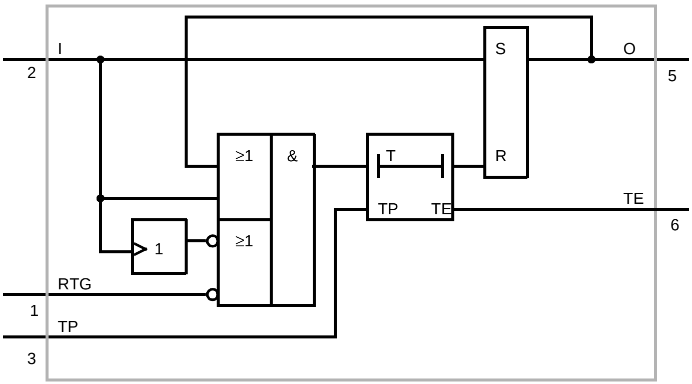
A memory is set when the input I is set. The output then goes to 1, see function diagram. When the time set in the timer has elapsed, the memory is cleared and the output O goes to 0.

MONO Function, Not Retriggerable

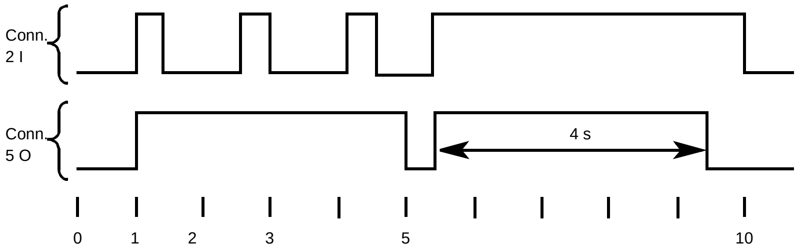
If the input RTG is reset, a MONO function which is not retriggerable is obtained. If a new pulse is obtained at the input I before the time set in the timer has elapsed, it does not affect the timer. Only when the time set has elapsed and the output O is reset can the MONO function be restarted by the input I going from 0 to 1.

MONO Function, Retriggerable

If RTG is set, a retriggerable MONO function is obtained, that is the timer starts from 0 each time a new pulse is obtained at the input I. If a new pulse is obtained at the input I before the time set in the timer has elapsed, the timer will restart, that is the MONO function is retriggerable.

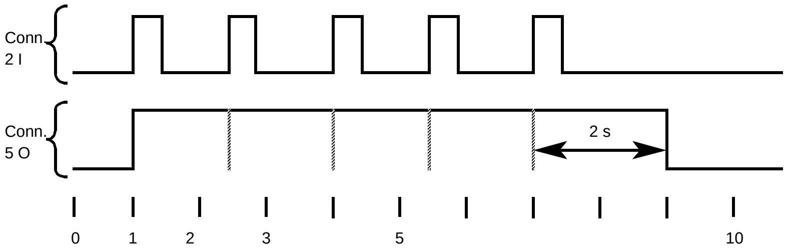


Function diagram



Conn.3 Time set=4 s
Conn.1 Reset

Time diagram MONO function, not retriggerable



Conn.3 Time set=2 s
Conn.1 Set

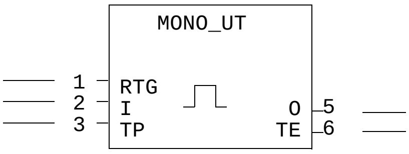
Time diagram MONO function, retriggerable

Monostable Untracked MONO_UT

Summary

The element MONO_UT (**MONO** function,-**UT** untracked) can be used like the function MONO, for example, for time limiting of operation of outputs and automatic functions. The element can also be used for impulse extension, stall alarm function, and so on.

In a redundancy configuration the output “O” is not tracked. In this configuration the PC element MONO_UT is used for independent life-signal supervision in a redundant MODBUS protocol



PC element MONO_UT

Call MONO_UT

Terminal Description

No	Name	Type	Description
1	RTG	IB	**R**e**T**rig**G**. Input for selection of if the MONO_UT function is to be retriggerable or not. If RTG is set, a retriggerable function is obtained.
2	I	IB	**I**nput. Input for start of the MONO_UT function when the input changes from 0 to 1.
3	TP	IT	**T**ime **P**ulse. Input for setting of pulse time.
5	O	OB	**O**utput. Output which is set when the MONO_UT function starts and resets (to 0) when the time set has elapsed. (Untracked in redundancy configuration)
6	TE	OT	**T**ime **E**lapsed. Output for time which has elapsed when O is set. When the time set (TP) has elapsed TE will stop.

Function

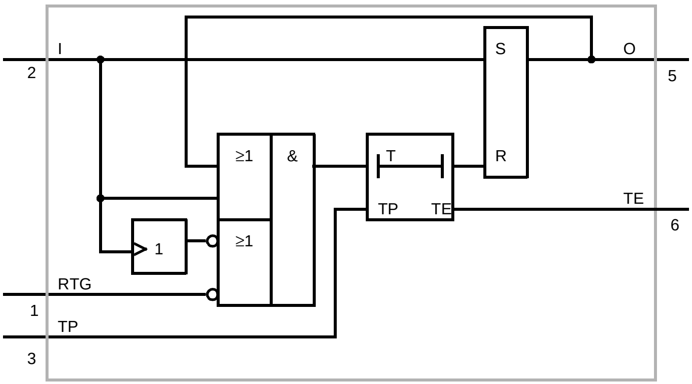
A memory is set when the input I is set. The output then goes to 1, see function diagram. When the time set in the timer has elapsed, the memory is cleared and the output O goes to 0.

MONO_UT Function, Not Retriggerable

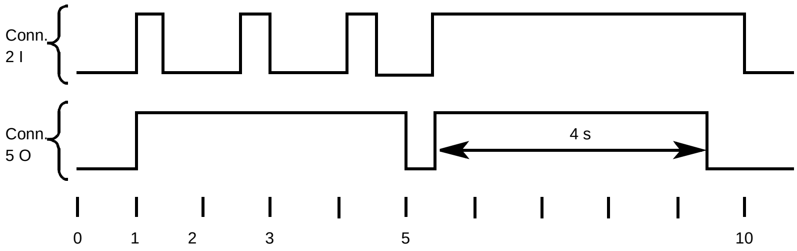
If the input RTG is reset, a MONO_UT function which is not retriggerable is obtained. If a new pulse is obtained at the input I before the time set in the timer has elapsed, it does not affect the timer. Only when the time set has elapsed and the output O is reset can the MONO_UT function be restarted by the input I going from 0 to 1.

MONO_UT Function, Retriggerable

If RTG is set, a retriggerable MONO_UT function is obtained, that is the timer starts from 0 each time a new pulse is obtained at the input I. If a new pulse is obtained at the input I before the time set in the timer has elapsed, the timer will restart, that is the MONO_UT function is retriggerable.

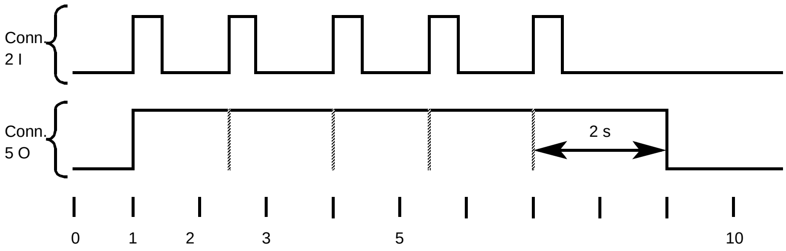


Function diagram



Conn.3 Time set=4 s
Conn.1 Reset

Time diagram MONO_UT function, not



Conn.3 Time set=2 s
Conn.1 Set

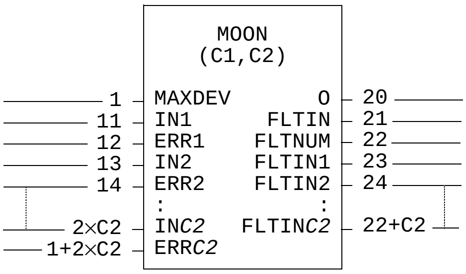
Time diagram MONO_UT function, retriggerable

M out of N **MOON**

Summary

The element MOON (**M** **O**ut **O**f **N**) is a function for supervision and selection of analog values in the case of set error bits or discrepancies between the analog values.

Disturbed inputs will be sorted out. Then, the remaining inputs will be recognized being undisturbed, if their deviation from the average value is smaller than the value at input MAXDEV. At least M values out of N inputs must be identified as undisturbed ones, that the measurement function will be considered being good.



PC element MOON

Call MOON (C1, C2)

Parameter	Significance	Permissible values
C1	Selection number M of inputs which at least must be undisturbed.	2…C2
C2	Number N of inputs IN1…INC2.	2…5

Terminal Description

No	Name	Type	Description
1	MAXDEV	IR	**MAX**imum permitted **DEV**iation between an input I and average value O. Default value 0.
2	IN1	IR	**IN**put for analog value **1**.
3	ERR1	IB	**ERR**or bit to input IN**1**. Default value 1.
4	IN2	IR	**IN**put for analog value **2**.
5	ERR2	IB	**ERR**or bit to input IN**2**. Default value 1.
: 2×C2	IN*C2*	IB	**IN**put for analog value **C2**.
1+2×C2	ERR*C2*	IB	**ERR**or bit to input IN**C2**. Default value 1.
20	O	OR	Average value **O**utput. O = Sum of undisturbed inputs/number of undisturbed inputs.
21	FLTIN	OB	**F**au**LT**, less than required number of undisturbed **IN**puts. FLTIN = 0 if at least M inputs are undisturbed (C1).
22	FLTNUM	OB	**NUM**ber of **F**au**LT**s, at least one disturbed input. FLTNUM = 0 if all inputs are undisturbed.
23	FLTIN1	OB	**F**au**LT**, input **IN1** is disturbed. FLTIN1 = 0 if the 1st input is undisturbed.

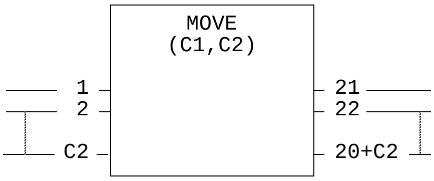
Terminal Description

No	Name	Type	Description
24	FLTIN2	OB	**F**au**LT**, input **IN2** is disturbed. FLTIN2 = 0 if the 2nd input is undisturbed.
: 22+C2	FLTIN*C2*	OB	**F**au**LT**, input **INC2** is disturbed. FLTINC2 = 0 if the input C2 is undisturbed.

Data Copying MOVE

Summary

MOVE copies the value at the input to the output and may be used to for example copy the value from one location in the data base to another.



PC element MOVE

Call MOVE (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	B, I, IL, R, T, TR
C2	Number of inputs and outputs	1…19

Terminal Description

No	Name	Type	Description
1	–	I*C1*	Input for value 1.
2	–	I*C1*	Input for value 2.
: C2	–	I*C1*	Input for value C2.
21	–	O*C1*	Output for value 1.
22	–	O*C1*	Output for value 2.
: 20+C2	–	O*C1*	Output for value C2.

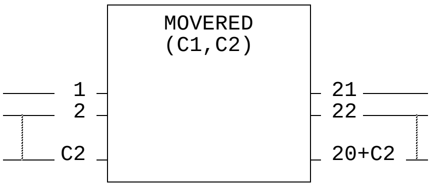
Function

The element copies the values at the inputs to the respective output. The type of the data to be moved is determined with the call parameter C1 and the number of values moved is determined with the call parameter C2.

Move Redundancy MOVERED

Summary

The element MOVERED (**MOVE** for **RED**undancy) is used with unplanned redundancies to copy the input value on the output value.



PC element MOVERED

Call MOVERED (C1, C2)

Parameter	Significance	Permissible values
C1	Data type of all inputs and outputs.	B, I, IL, R, T, TR
C2	Number of inputs and outputs.	1…19

Terminal Description

No	Name	Type	Description
1	–	I*C1*	Input value 1.
2	–	I*C1*	Input value 2.
: C2	–	I*C1*	Input for value C2.
21	–	O*C1*	Output value 1.
22	–	O*C1*	Output value 2.
: 20+C2	–	O*C1*	Output for value C2.

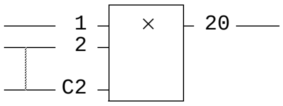
Function

With an unplanned redundancy, when outputs are recursively connected to the inputs, these feedbacks must be planned. For this purpose, the element MOVERED must be connected in the line between the input and output.

Multiplier MUL

Summary

MUL (**MUL**tiplier) is used for multiplication of an optional number of integers or real numbers.



PC element MUL

Call MUL (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R
C2	Number of inputs	2…19

Terminal Description

No	Name	Type	Description
1	I*C1*		Input for multiplicand
2	I*C1*		Input for multiplier
: C2	I*C1*		Input for multiplier
20	O*C1*		Output for product

Function

The values at the inputs 1…C2 are multiplied and the product is stored at the output 20.

Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

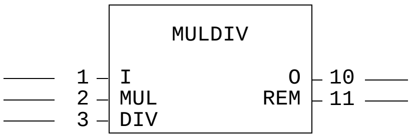
Integer Scaling Element

MULDIV

Summary

MULtiplier **DIV**ider is used to scale integer values by dividing the product of two integer inputs.

Call MULDIV



PC element MULDIV

Terminal Description

No	Name	Type	Description
1	I	II	**I**nput for multiplicand
2	MUL	II	Input for **MUL**tiplier
3	DIV	II	Input for **DIV**isor
10	O	OI	**O**utput for quotient
11	REM	OI	Output for **REM**inder

Function

The product of input I and input MUL is divided by input DIV. The quotient is loaded at the output O and the reminder at REM. The element internally uses 32 bit accuracy to perform the multiplication.

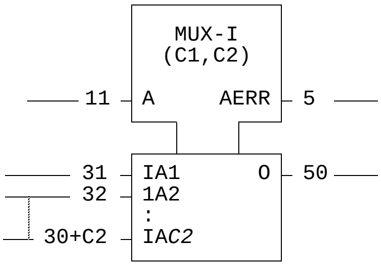
Overflow

If the maximum positive value is exceeded, then the output O is limited to +32767. If the minimum negative value is exceeded, then the output O is limited to -32767.

Multiplexer with Integer Address MUX-I

Summary

MUX-I (**MU**ltiple**X**er **-** with **I**nteger address) is used as a selector and has an optional number of inputs. The data type can be integers, real numbers, Boolean or time.



PC element MUX-I

Call MUX-I (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	2…19

Terminal Description

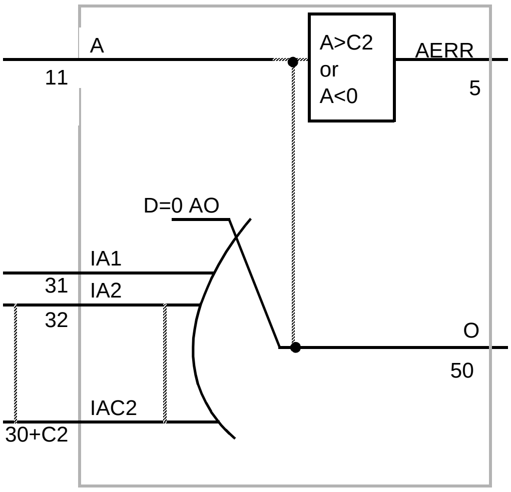
No	Name	Type	Description
5	AERR	OB	**A**ddress **ERR**or. Output which is set when the address value is larger than the number of inputs, or negative.
11	A	II	**A**ddress. Input for address value which specifies which input is to be connected to the output O. When A = 0, the output is set to 0.
31	IA1	I*C1*	**I**nput **A**ddress **1**. The second input to the selector.
32	IA2	I*C1*	**I**nput **A**ddress **2**. The second input to the selector.
: 30+C2	IA*C2*	I*C1*	**I**nput **A**ddress ***C2***. The last input to the selector.
50	O	O*C1*	**O**utput. Data output from the selector.

Function, Addressing

The input, the data value (IA1…IA*C2*) of which is to be connected to the output O, is specified with an address (integer 1…C2) at the input A. If the address is 0, the output terminal O is set to zero.

Supervision

The address A is monitored and if its value is greater than the number of inputs or is negative, the error signal output AERR is set. The data value 0 is then obtained at the output.

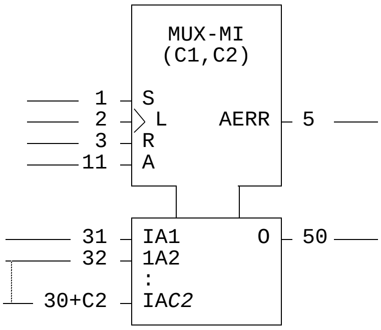


Function diagram

Multiplexer with Memory and Integer Address MUX-MI

Summary

MUX-MI (**MU**ltiple**X**er **-** with **M**emory and **I**nteger address) is used as a selector with memory function and has up to 19 inputs. The data type can be integers, real numbers, Boolean or time.



PC element MUX-MI

Call MUX-MI (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	1…19

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for loading of new values each time the element is executed. When the input S is reset 0, the last value loaded will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of the data addressed.
3	R	IB	**R**eset. Input for clearing of the outputs. This input overrides the S and L inputs.
5	AERR	OB	**A**ddress **ERR**or. Output which is set when the address value is greater than the number of inputs or is negative.
11	A	II	**A**ddress. Input for address value which specifies from which input data is to be loaded. When A=0, the output is reset to 0.
31	IA1	I*C1*	**I**nput **A**ddress **1**. The first input to the selector.
32	IA2	I*C1*	**I**nput **A**ddress **2**. The second input to the selector.
: 30+C2	A*C2*	I*C1*	**I**nput **A**ddress ***C2***. The last input to the selector.
50	O	O*C1*	**O**utput. Data output from the selector.

Function, Addressing

The input data value (IA1…IA*C2*) that is to be stored at the output is specified with an address (integer 1…C2) at the input A. If the address is 0, the data value 0 is stored.

Loading

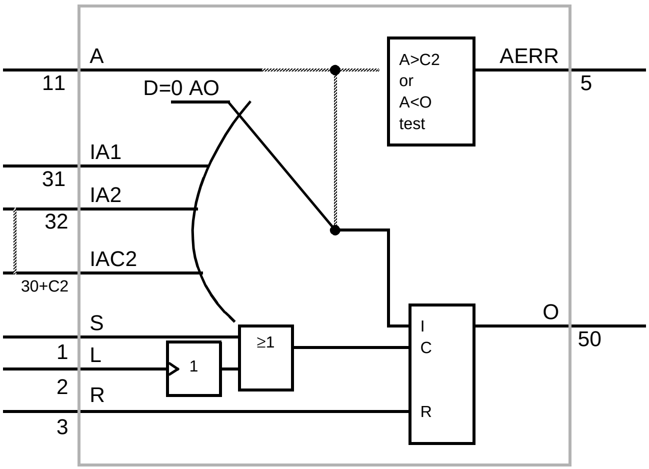
The value is loaded at the instant the input L is set to 1. If input S is set, new data is loaded each time the element is executed. When S is reset after having been set, the data loaded most recently remains until the element is once more executed with input S, L or R set. Input S overrides input L, that is when S is set, L has no function.

Clearing

The input R clears the data output and prevents further storage of data.

Supervision

The address A is monitored and if its value is greater than the number of inputs or is negative, the error signal output AERR is set. The data value 0 is then stored at the output.

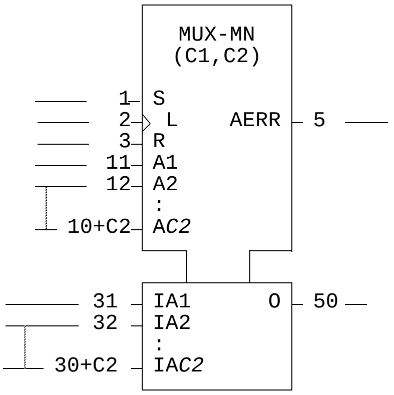


Function diagram

Multiplexer with Memory and 1 of N Address MUX-MN

Summary

MUX-MN (**MU**ltiple**X**er **-** with **M**emory and 1-of-**N** address) is used as a selector with memory function. MUX-MN has up to 19 inputs. The data type can be integers, real numbers, Boolean or time.



PC element MUX-MN

Call MUX-MN (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	1…19

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for loading of new values each time the element is executed. When the input S is reset, the last value loaded will remain at the outputs.
2	L	IB	**L**oad. Dynamic input for loading of the data addressed.
3	R	IB	**R**eset. Input for clearing of the outputs. This input overrides the S and L inputs.
5	AERR	OB	**A**ddress **ERR**or. Output which is set when two or more of the inputs A1…A*C2* are set.
11	A1	IB	**A**ddress **1**. Input to store data from the input IA1 at the output O.
12	A2	IB	**A**ddress **2**. Input to store data from the input IA1 at the output O.
10+C2	A*C2*	IB	**A**ddress ***C2***. Input to store data from the input IA*C2* at the output O.
31	IA1	I*C1*	**I**nput **A**ddress **1**. The first input to the selector.
32	IA2	I*C1*	**I**nput **A**ddress **2**. The second input to the selector.
: 30+C2	IA*C2*	I*C1*	**I**nput **A**ddress *C2*. The last input to the selector.
50	O	OC1	**O**utput. Data output from the selector.

Function, Addressing

The input data value (A1…IA*C2*) that is to be stored at the output is specified with the inputs A1…A*C2*. If the input A1 is 1, the value from input IA1 is stored at the output, if A2 is 1, the value from input IA2, and so on. If none of the inputs A1…A*C2* is set to 1, the data value 0 is stored at the output O.

Loading

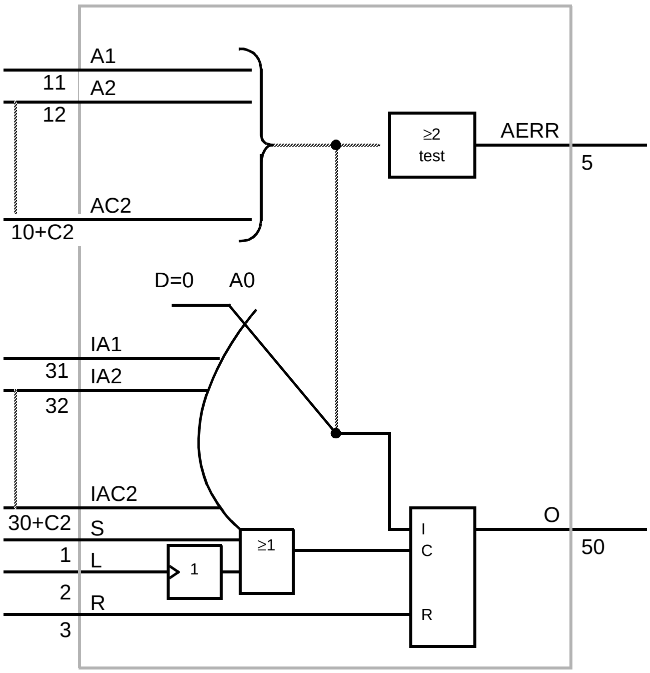
The value is loaded at the instant the input L is set to 1. If input S is set, new data is loaded each time the element is executed. When S is reset after having been set, the data loaded most recently remains. Input S overrides input L, that is when S is set, L has no function.

Clearing

The input R clears the data output and prevents further storage of data.

Supervision

If two or more of the inputs A1…A*C2* are set simultaneously, the output AERR is set and the value from the input which correspond to the address input set with the lowest number is stored at the output O.

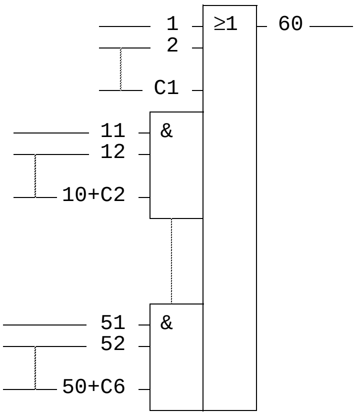


Function diagram

Or Gate with And Inputs OR-A

Summary

OR-A (**OR -** with **A**nd gates on the inputs) is used to form general combined expressions with Boolean variables.



PC element OR-A

Call OR-A (C1, C2, C3, …, Cn)

Parameter	Significance	Permissible values
C1	Number of inputs in the OR part of the element	0…9
C2	Number of inputs in the first AND gate of the element	0…9
C3	Number of inputs in the second AND gate of the element	0…9
: C*n*(1)	Number of inputs in the last AND gate of the element	0…9

(1)1≤*n*≤6, 1≤C1+C2+...+C*n*≤40

Terminal Description

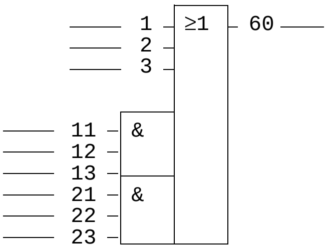
No	Name	Type	Description
1	–	IB	Input direct to the OR gate in the element.
2	–	IB	Input direct to the OR gate in the element.
: C1	–	IB	Input direct to the OR gate in the element.
11	–	IB	Input to the first AND gate in the element.
12	–	IB	Input to the first AND gate in the element.
10+C2	–	IB	Input to the first AND gate in the element.

Terminal Description (Continued)

No	Name	Type	Description
21	–	IB	Input to the second AND gate in the element.
22	–	IB	Input to the second AND gate in the element.
: 20+C3	–	IB	Input to the second AND gate in the element.
10×(n-1)+1	–	IB	Input to the last AND gate in the element.
10×(n-1)+2	–	IB	Input to the last AND gate in the element.
: 10×(n-1)+Cn	–	IB	Input to the last AND gate in the element.
60	–	OB	Output.

Function

More rapid execution than if separate elements are used can be obtained by using assembled elements such as OR-A. The inputs need then only be tested until the value of the output can be determined. If input 1 is 1, the output can be set irrespective of the other inputs. See the figure and the table below.



PC element OR-A (3, 3, 3)

Table OR-A (3, 3, 3)

1	2	3	11	12	13	21	22	23	60
0	0	0	0	0	0	1	1	1	1
0	0	0	0	×	×	0	×	×	0
0	0	0	0	×	×	1	0	×	0
0	0	0	0	×	×	1	1	0	0
0	0	0	1	0	×	0	×	×	0
0	0	0	1	0	×	1	0	×	0
0	0	0	1	0	×	1	1	0	0
0	0	0	1	1	0	0	×	×	0
0	0	0	1	1	0	×	0	×	0

Table OR-A (3, 3, 3)

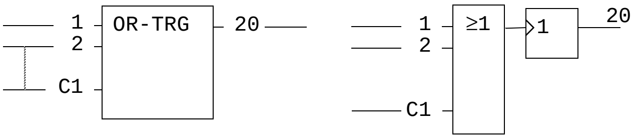
1	2	3	11	12	13	21	22	23	60
0	0	0	1	1	0	×	×	0	0
0	0	0	1	1	1	×	×	×	1
0	0	1	×	×	×	×	×	×	1
0	1	×	×	×	×	×	×	×	1
1	×	×	×	×	×	×	×	×	1

× indicates that the input has no effect on the value of the output or the execution time.

Or Gate With Trigger on Outputs OR-TRG

Summary

OR-TRG (**OR** - with **TR**i**G**ger on the output) is used when the output of an OR gate should be triggered.



PC element OR-TRG

Call AND-TRG (C1)

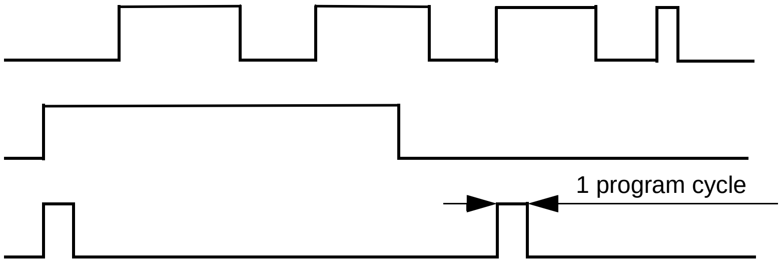
Parameter	Significance	Permissible values
C1	Number of OR inputs	2…19

Terminal Description

No	Name	Type	Description
1	–	IB	Input to the OR gate.
2	–	IB	Input to the OR gate.
: C1	–	IB	Input to the OR gate.
20	–	OB	Output

Function

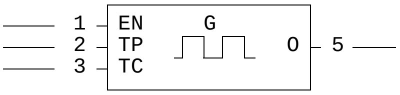
When the result of the OR gate is set (to 1) the output signal is also set. The output signal is cleared during the next program cycle, irrespective of the result of the OR gate. For the output signal to be set, the result of OR gate must have been 0 during one program cycle. The output signal from the element may only be used within its own execution unit as detection of it is otherwise not certain.



Oscillator OSC-B

Summary

Oscillator OSC-B (**OSC**illator **- B**oolean variables) with variable frequency and pulse time is used when pulse trains with periods from 2× cycle time up to 24 hours are needed.



PC element OSC-B

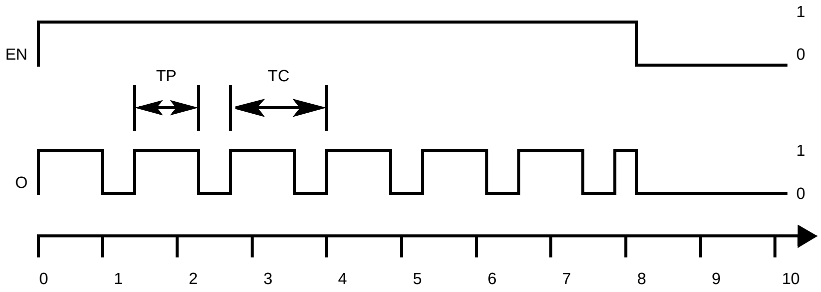
Call OSC-B

Terminal Description

No	Name	Type	Description
1	EN	IB	**EN**able. Input for start of oscillator.
2	TP	IT	**T**ime **P**ulse. Input for setting of pulse time.
3	TC	IT	**T**ime **C**ycle. Input for setting of period. The period must be greater than the pulse duration.
5	O	OB	**O**utput. Oscillator output.

Function

The oscillator starts when the input EN is set (to 1). The figure below shows an example of an oscillator time diagram.

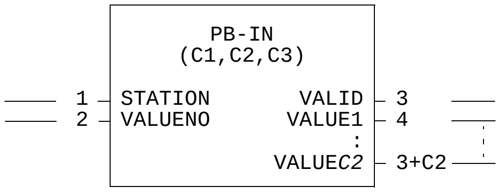


Example of time diagram with TP=1 s and TC=1.5 s

Input from PROFIBUS PB-IN

Summary

PC element PB-IN is used to transfer process or diagnostic values from the PROFIBUS slaves to Advant Controller 160.



PC element PB-IN

Call PB-IN (C1, C2, C3)

Parameter	Significance	Permissible values
C1	Data type	B, I, IL, R
C2	Number of inputs	1…32 (C1=B C3=0) 1…8 (C1=B C3=1) 1 (C1=I, IL, R C3=0)
C3	Process/Diagnostic data kind selector	0 = Process data 1 = Diagnostic data(1)

(1) For further information please look into the PROFIBUS specification

Terminal Description

No	Name	Type	Description
1	STATION	II	**STATION** number representing the slave on the PROFIBUS. Range 0…125
2	VALUENO	II	**VALUE N**umber of the PROFIBUS slave. Range 1…244 (if C3 = 1, then 245 represents the internal PROFIBUS interface slave status).
3	VALID	OB	**VALID**. 1 = Data has been updated, 0 = Data has not been received.
4	VALUE1	OC1	**VALUE1** of type B, I, IL, R.
5	VALUE2	OC1	**VALUE2** of type B.
: 3+C2	VALUEC2	OC1	**VALUEC2** of type B.

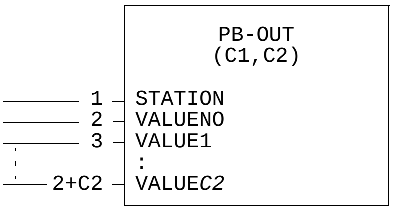
Function

PC element PB-IN is used to transfer process or diagnostic values from the PROFIBUS slaves to Advant Controller 160. The call parameter are used to select number, type, and kind of values.

Output to PROFIBUS PB-OUT

Summary

PC element PB-OUT is used to transfer values from Advant Controller 160 to the PROFIBUS slaves.



PC element PB-OUT

Call PB-OUT (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	B, I, IL, R
C2	Number of outputs	1…32 (C1=B) 1 (C1=I, IL, R)

Terminal Description

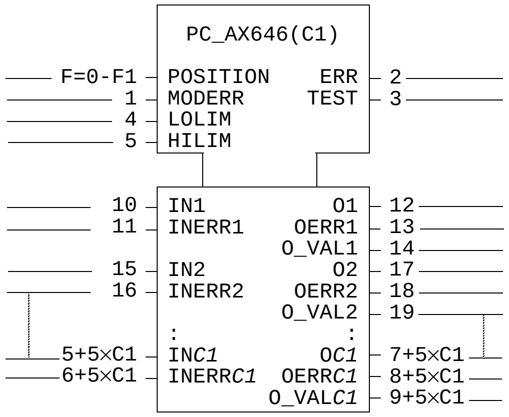
No	Name	Type	Description
1	STATION	II	**STATION** number representing the slave on the PROFIBUS. Range 0…125
2	VALUENO	II	**VALUE N**umber of the PROFIBUS slave. Range 1…244.
3	VALUE1	IC1	**VALUE1** of type B, I, IL, R.
4	VALUE2	IC1	**VALUE2** of type B.
: 3+C2	VALUEC2	IC1	**VALUEC2** of type B.

Function

PC element PB-OUT is used to transfer values from Advant Controller 160 to the PROFIBUS slaves. The call parameter are used to select number, type, and kind of values.

Operating and testing AX646 PC_AX646

Summary



PC element PC_AX646

AX646 is basically an analog input module providing 16 analog input channels for current measurement 4…20 mA. Furthermore the device has 16 analog outputs which can be connected inside the module to the related inputs for testing. The output signals are not available to the user. The device can be switched into test mode by a binary signal given from the Processor Module via BIOB. The purpose of this PC Element is to ease the engineering of an AX646.

The PC Element does all the testing of the module AX646. The user has access to the outputs of this PC Element like the outputs of the DB element AX646. In contrast to the DB element AX646 all channels are tested by the PC Element. The input tests of AX646 are done alternately. After 20 times of reading process values with the cycle time of the control module (CONTRM) the module will be switched into test mode for one cycle. In this test cycle the previous written output values will be read back and compared. If the difference between the test output value and read input value is greater than 2% the ERR terminal of the signal DB element is set to 1 and the output of the PC Element is set to the last process value. The PC Element is automatically reset if all channels are faulty for 50 cycles and no channel is disturbed after this time. The minimum cycle time of the CONTRM is 100 ms. The test includes linearity, gain, offset, cross-talking of the analog/digital converter, the digital/analog converters and the multiplexer. The complete input test is done after 6 test cycles. Each test cycle has a set of predefined test pattern. The test will find 90% of all possible errors in the module.

Call PC_AX646(C1)

Parameter	Significance	Permissible values
C1	Number of used channels	1…16

Terminal Description

No	Name	Type	Description
F1	POSITION	F	**POSITION** of corresponding AX646 device. 1–20, 101–120, 201–220, …, 701–720; but not ×16
1	MODERR	IB	**MOD**ule **ERR**or. This terminal will be connected to the ERR terminal of the module. (e.g. AXj:ERR, where j is the DB instance of the module.

Terminal Description (Continued)

2	ERR	OB	**ERR**or. Output which is set when one or more OERRx Bits are set to 1. (Sum error of all OERR). Also set if POSITION is invalid.
3	TEST	OB	**TEST** output must be connected to the data base terminal AXx.33 of the corresponding DB element AX646.
4	LOLIM	IR	**LO**w **LIM**it. The value can be set to -25 …118%.
5	HILIM	IR	**HI**gh **LIM**it. The value can be set to -25% …118%.
10	IN1	IIL	**IN**put **1**. This terminal will be connected to the value terminal of the first channel.
11	INERR1	IB	**IN**put **ERR**or **1**. This terminal will be connected to the ERR terminal of the first channel.
15	IN2	IIL	**IN**put **2**. This terminal will be connected to the value terminal of the second channel.
16	INERR2	IB	**IN**put **ERR**or **2**. This terminal will be connected to the ERR terminal of the second channel.
: 5+(C1×5)	IN*C1*	IIL	**IN**put ***C1***. This terminal will be connected to the value terminal of the C1 channel.
6+(C1×5)	INERR*C1*	IB	**IN**put **ERR**or ***C1***. This terminal will be connected to the ERR terminal of the C1channel.
12	O1	OR	**O**utput **1**. First Process Value. This value will be used in PC program. O1 and OERR1 must not be used at the same channel.
13	OERR1	OB	**O**utput **ERR**or channel **1**. ERR of the first process value. This is the result of testing and self-test of the AX646.
14	O_VAL1	OR	**O**utput Test_**VAL**ue channel **1**.
17	O2	OR	**O**utput **2**. First Process Value. This value will be used in PC program. O1 and OERR1 must not be used at the same channel.
18	OERR2	OB	**O**utput **ERR**or channel **2**. ERR of the second process value. This is the result of testing and self-test of the AX646.
19	O_VAL2	OR	**O**utput Test_**VAL**ue channel **2**.
: 7+(C1×5)	OC1	OR	**O**utput ***C1***. Process Value channel C1. This value will be used in PC program. OC1 and OERRC1 must not be used at the same channel.
8+(C1×5)	OERRC1	OB	**O**utput **ERR**or channel ***C1***. ERR of the process value channel C1. This is the result of testing and self-test of the AX646.
9+(C1×5)	O_VALC1	OR	**O**utput Test_**VAL**ue channel **C1.**

Function

This element copies the value at input IN*i* to output O*i*. OERR*i* is the result of an “OR” combination of the ERR terminals of all signal DB elements and test of the AX646 element. Unused channels should not be connected to the PC Element. The cycle time of the CONTRM for this PC Element must not be shorter than 100 ms.

Connection

MODERR will be connected to AX*j*:ERR

IN*i* will be connected to AXI*j*.*i*:VALUE

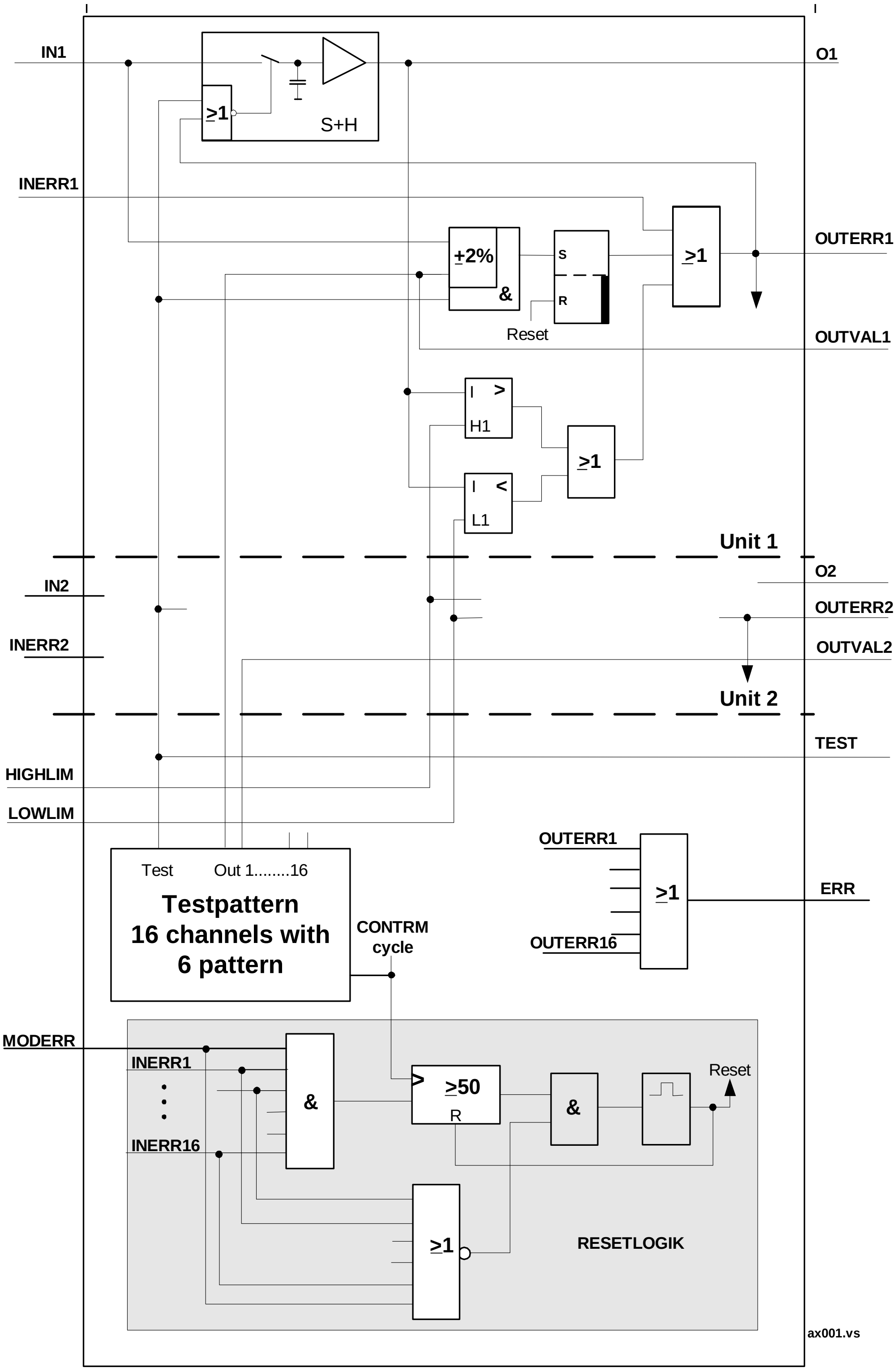
INERR*i* will be connected to AXI*j*.*i*:ERR

where j = module number, i = channel number.

Note

Do not connect channel *i* of the AX646 DB element to IN*j* of the PC Element if i ≠ j).

Diagram of internal function of the PC Element



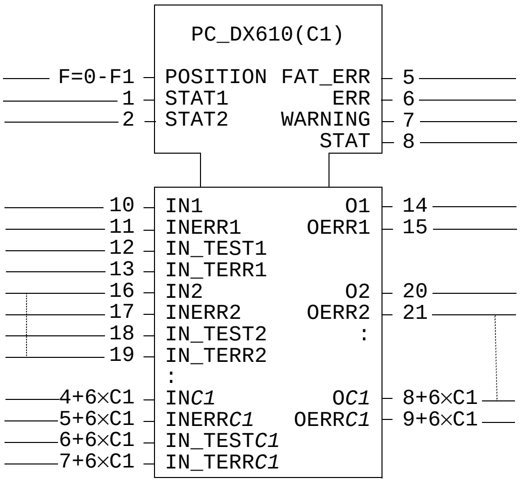
Operating and Testing DX610 PC_DX610

Summary

The device DX610 is basically a digital output module providing 6 digital output channels with relays. The output signals could be read back to test the device. The PC Element PC_DX610 does all the testing of the module DX610 for safety application. The testing is twofold. Every cycle the PC Element checks whether the read back input signal is the same as the output signal. If not this channel is set into the error state and the output signal is put to zero. In a 2 out of 3 application at least every 24 hour the PC Element tests “stack to high” and cross-talking. If an error is found the output is set to zero and the corresponding error is put to one. “Stack to low” is never tested.

This PC Element also supervises the 2 out of 3 stations. If the corresponding 2 out of 3 channels are different or have an error for more than 24 hour this specific channel is set to zero. In case of a serious error of the Processor Module DB element the PC element will switch all outputs to zero.

The minimum cycle time of the control module (CONTRM) in PM64x is 50 ms. The test will find 90% of all possible errors in the module.



PC element PC_DX610

Call PC_DX610(C1)

Parameter	Significance	Permissible values
C1	Number of used channels	1…6

Terminal Description

No	Name	Type	Description
F1	POSITION	F	**POSITION** of DX610 Device 1–20, 101–120, 201–220, …, 701–720.
1	STAT1	II	This Input need to be connected to the corresponding PC_DX610 **STAT** output in the other station. The communication will take place via upper HSL (**1**).
2	STAT2	II	This Input need to be connected to the corresponding PC_DX610 **STAT** output in the other station. The communication will take place via lower HSL (**2**).
5	FAT_ERR	OB	The second shutdown path was activated by this PC Element because of a serious hardware error.
6	ERR	OB	Sum **ERR**or of all error terminals. Also set if POSITION is invalid, or if non-redundant Processor Modules are used.
7	WARNING	OB	**WARNING** indication if the value of any of the 2 out of 3 channels are different.

Terminal Description (Continued)

8	STAT	OI	**STAT**us of the values and errors. It should be connected to STAT1 or STAT2 in the other stations.
10	IN1	IB	This terminal will be connected to the **IN**put value coming from the PC part. This will be the desired output value of the DX610 channel **1**.
11	INERR1	IB	**ERR**or Terminal of the **IN1** value.
12	IN_TEST1	IB	Value connected to the DXi.7:ERR terminal of the DX610 for check back.
13	IN_TERR1	IB	**ERR**or terminal of the **IN**_**T**EST**1**.
16	IN2	IB	This terminal will be connected to the **IN**put value coming from the PC part. This will be the desired output value of the DX610 channel **2**.
17	INERR2	IB	**ERR**or Terminal of the **IN2** value.
18	IN_TEST2	IB	Value connected to the DXi.8:ERR terminal of the DX610 for check back.
19	IN_TERR2	IB	**ERR**or terminal of the **IN**_TEST**2**.
: 4+6×C1	IN*C1*	IB	This terminal will be connected to the **IN**put value coming from the PC part. This will be the desired output value of the DX610 channel ***C1***.
5+6×C1	INERR*C1*	IB	**ERR**or Terminal of the **IN*C1*** value.
6+6×C1	IN_TEST*C1*	IB	Value connected to the DXi.C1+6:ERR terminal of the DX610 for check back.
7+6×C1	IN_TERR*C1*	IB	**ERR**or terminal of the **IN**_TEST***C1***.
14	O1	OB	**O**utput value. This value will be connected to DXi.**1**
15	OERR1	OB	**ERR**or of **O**utput channel **1**. This is the result of testing and self-test of the DX610.
20	O2	OB	**O**utput value. This value will be connected to DXi.**2**
21	OERR2	OB	**ERR**or of **O**utput channel **2**. This is the result of testing and self-test of the DX610.
: 8+6×C1	O*C1*	OB	**O**utput value. This value will be connected to DXi.***C1***
9+6×C1	OERR*C1*	OB	**ERR**or of **O**utput channel ***C1***. This is the result of testing and self-test of the DX610.

Function

This element copies the value at input IN*i* to output O*i*. OERR*i* is the result of an “OR” combination of the ERR terminals of all signal DB elements and test of the DX610 element.

Connection

O*j* will be connected to DX*i*.*j* (*i* = module number, *j* = channel number)

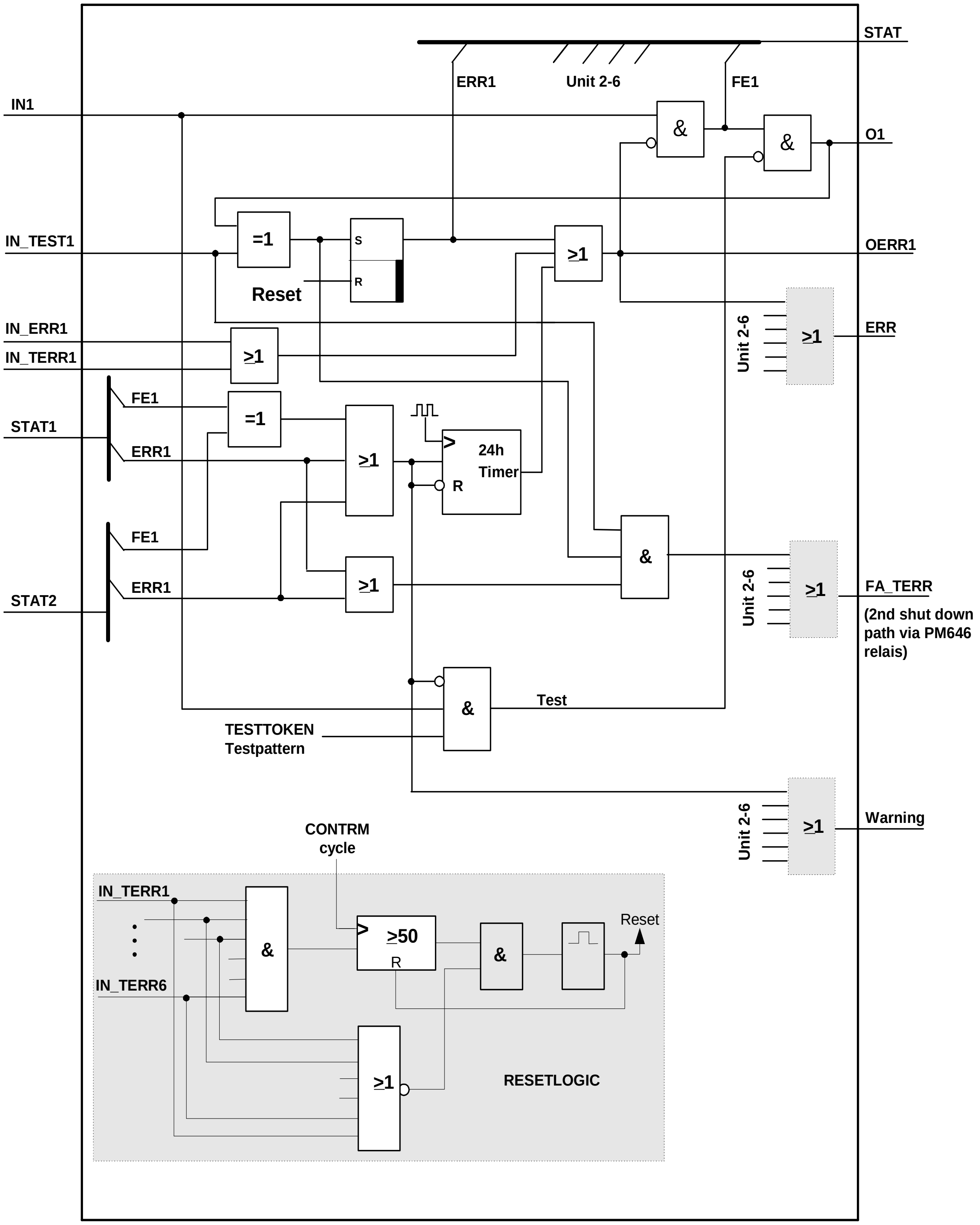
IN_TEST*j* will be connected to DX*i*.*j*+6:ERR (*i* = module number, *j* = channel number).

Note

Do not connect channel *i* of the DX610 DB element to IN*j* of the PC Element if $i \neq j$).

PC_DX610 is designed for use in redundant applications. Do not use it for non-redundant applications.

Diagram of Internal Function

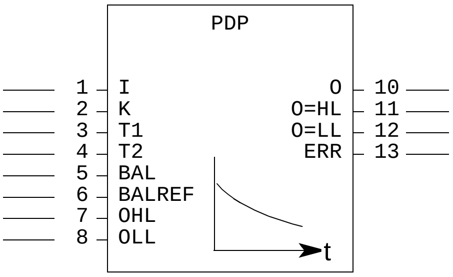


PDP-Function

PDP

Summary

PDP (**P**roportional **D**erivative **P**roportional function) is used to give proportional effect and limited derivation effect.
The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from balancing or from limited output signal are bumpless.



PC element PDP

Call PDP

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	T1	ITR	**T**ime **1**. Input for time constant for lead.
4	T2	ITR	**T**ime **2**. Input for time constant for lag.
5	BAL	IB	**BAL**ance. Input for activation of following.
6	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
7	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
8	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput = **L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.

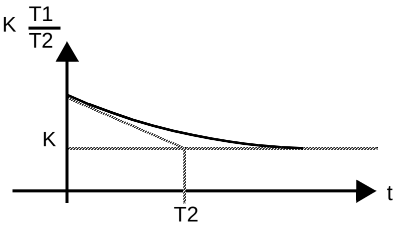
Function

The step response for a PDP function is:

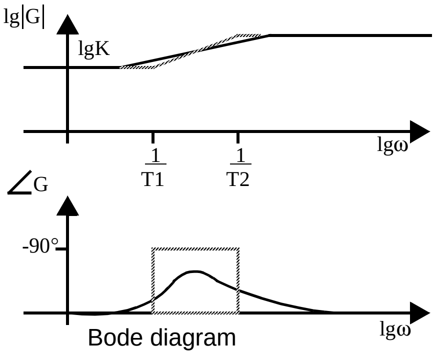
$O(t) = I(t) \times (K + k1e^{-t/T2})$ where $k1 = K(T1/T2 - 1)$

The transfer function for a PDP function is:

$G(s) = K(1+sT1)/(1+sT2)$ where $T2 < T1$



Step response

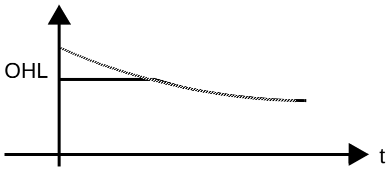


Bode diagram

This has been implemented in the PDP element as a recursive algorithm. The design of this algorithm is such that normal functioning is maintained even during limiting. This ensures a controlled return to a dynamic state.

Gain, Time Constants and Sampling Time

Certain constants are precalculated to make the execution time of the element as short as possible. The results are stored internally in the element. These constants are recalculated if T1, T2 or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating a test is performed to check whether T1 and $T2 \geq 2 \times TS$. If this is not the case T1 or T2 respectively is set equal to $2 \times TS$.



Function in limiting state

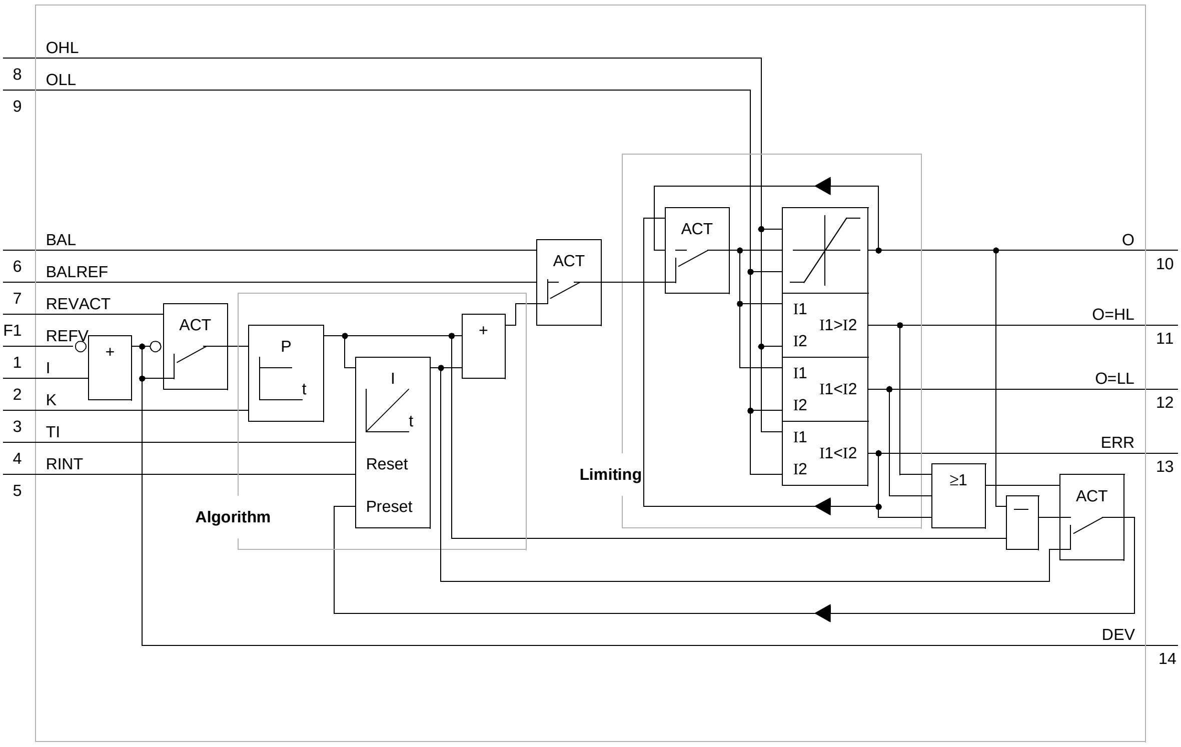
Following

If BAL is set to 1, the regulator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limits value concerned. Return to dynamic state is bumpless.

Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for the upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1.

The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the value they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following.



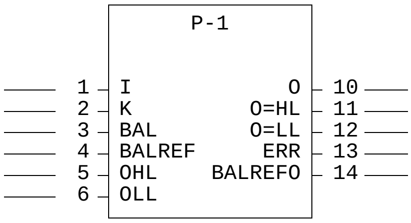
Function diagram

Proportional Regulator P-1

Summary

P-1 (**P**roportional Function) is used as an amplifier in feedback systems. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference.

Call P-1



PC element P-1

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input signal.
2	K	IR	Input for setting gain.
3	BAL	IB	**BAL**ance. Input for activation of following.
4	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
5	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
6	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output is limited to the upper limit value.
12	O=LL	OB	**O**utput = **L**ow **L**imit. Output which is set to 1 if the output is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output which is set to 1 if OHL is less than OLL.
14	BALREFO	OR	**BAL**ance **REF**erence **O**utput. Information output for O/K.

Function

The P-function can be written in the time plane as $O(t) = K\,I(t)$. Its main property when controlling is that the output signal is directly proportional to the input signal. The transfer function for a P-function is: $G(S) = K$

Following

If BAL is set to 1, the amplifier goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned.

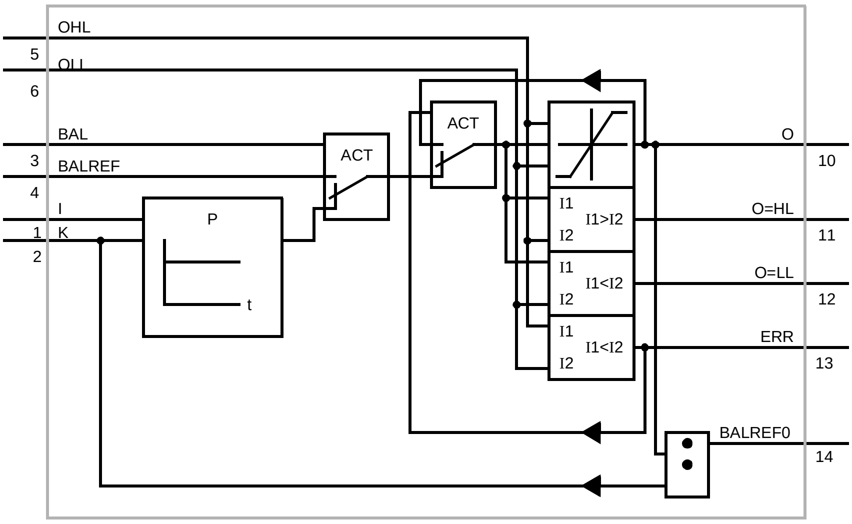
Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit value and OLL for the lower limit value. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1.

The element checks that the upper limit OHL is greater than the lower limit OLL. If not, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred.

Information Output BALREFO

The output BALREFO gives the result of the calculation O(t)/K where O(t) is the output signal after the limitation. The output specifies the input signal which corresponds to the output signal concerned.

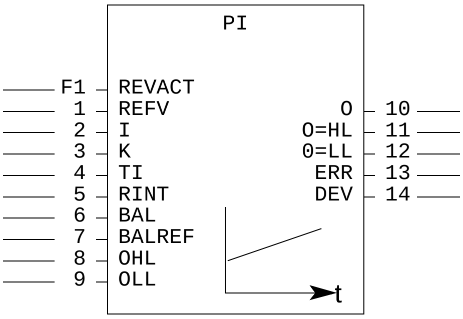


Function diagram

PI-Regulator PI

Summary

PI (**P**roportional **I**ntegrating regulator) is used as a standard PI- regulator for serial compensation in feed back systems. The control deviation is calculated internally in the element. The output signal can be limited to limits specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to normal function. All transfers from balancing or limited output signal are bumpless.



PC element PI

Call PI

Terminal Description

No	Name	Type	Description
F1	REVACT	IB	**REV**erse **ACT**ion. Parameter input for inverted control action. REVACT=1 gives inverted control action.
1	REFV	IR	**REF**erence **V**alue. Input for desired value.
2	I	IR	**I**nput. Input for actual value.
3	K	IR	Input for setting gain.
4	TI	ITR	**T**ime **I**ntegration. Input for time constant for integration.
5	RINT	IB	**R**eset **INT**egrator. Input for clearing of integrator.
6	BAL	IB	**BAL**ance. Input for activation of following.
7	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
8	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit.
9	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput = **L**ow **L**imit. Output which is set to 1 if the output is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.
14	DEV	OR	**DEV**iation. Output for control deviations, that is DEV=1–REFV.

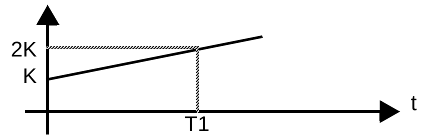
Function, Control Deviation

DEV, the control deviation, is calculated as follows even when the regulator is following: DEV = I–REFV

Inverted Control Effect

REVACT is a function parameter which controls the direction of the output signal change in relation to the actual value.

If REVACT is set to 0, direct action is obtained, that is increasing actual value gives increasing output signal. When REVACT is set to 1, inverted action is obtained, that is increasing actual value gives decreasing output signal.



Step response

Transfer Function

The PI function can be written in the time plane as

$O(t) = K(DEV(t)+\int^{\tau} DEV(\tau)/TI\, d\tau)$

and its main property when controlling is that it permits the integral section to retain its value when the error DEV(t)=0, that is the output signal is then constant.

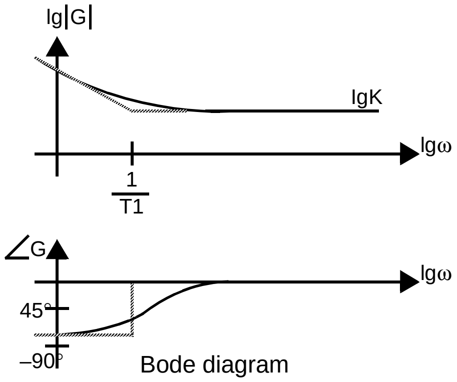
The transfer function for a PI function is:

$G(s) = K(1+1/s \times TI)$

This has been implemented in the PI element as a recursive algorithm.

Gain, Integration Time Constant and Sampling Time

The constant K×TS/TI is precalculated to make the execution time of the element as short as possible. The result is stored internally in the element. This constant is recalculated if TI or K is changed more than 1/128 of the preceding value or if the sampling time TS is changed. When the value is recalculated a test is performed to ensure that TS/TI<1. If not, TS/TI is set to 1.



Bode diagram

Clearing of Integrator

The integration section of the algorithm is cleared when RINT goes to 1. If the proportional section of the algorithm exceeds the limits of the output signal, the limit status remains and the internal status is updated in accordance with the section “Bumpless transfer from following or limitation”. When RINT is set permanently to 1, the element functions as a P-regulator.

Following

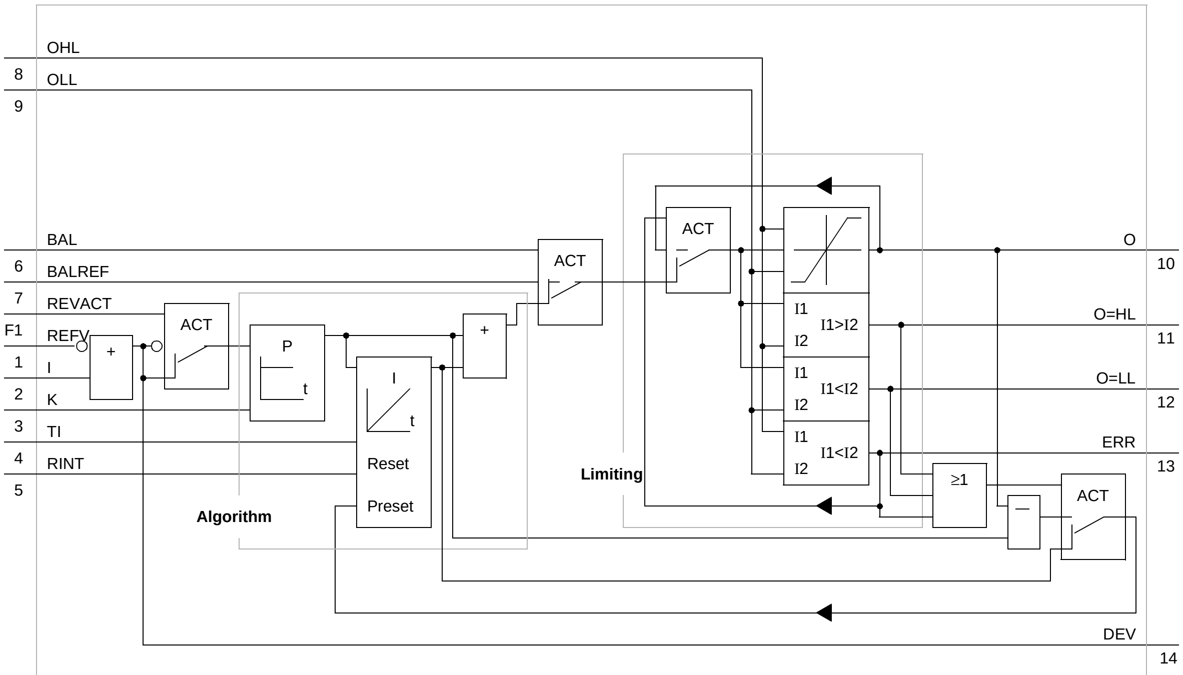
If BAL is set to 1, the regulator immediately goes into following and the output O follows the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. On return to a normal function the value of the output O during the last sample in the following remains during one sample time. See below under “Bumpless transfer from following or limiting”.

Limitation Function

The limitation function limits the output signal to the limit values at the inputs OHL for upper limit value and OLL for the lower limit value. If the actual value exceeds the upper limit value, the output O=HL is set to 1 and if it falls below the lower limit value, the output O=LL is set to 1. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1. While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred.

Bumpless Transfer from Following or Limitation

Transfer from following status (BAL=1) or from a limited output signal is bumpless. This is performed by recalculation of internal states, that is the integration part according to $INT(t) = O(t) - K \times DEV(t)$.

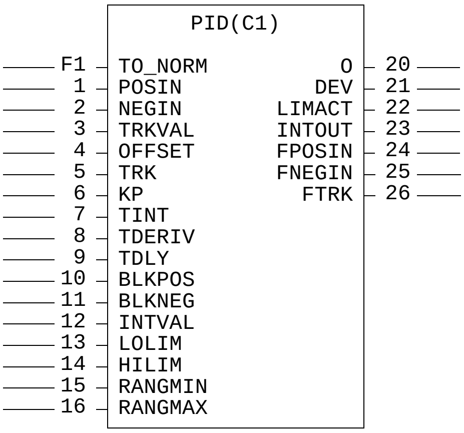


Function diagram

PID-Algorithm PID

Summary

The element PID (**PID**-Algorithm) implements a PID controller and is used to create simple control loops on the function group and single-variable level.



PC element PID

Call PID (C1)

Parameter	Significance	Permissible values
C1	Not used.	1

Terminal Description

No	Name	Type	Description
F1	TO_NORM	IB	Switch for: Standardization **TO NORM**al output signal. Default value 0.
1	POSIN	IR	**POS**itive **IN**put.(1)
2	NEGIN	IR	**NEG**ATIVE **IN**put.(1)
3	TRKVAL	IR	**TR**ac**K**ing **VAL**ue for integrator.
4	OFFSET	IR	**OFFSET**. Disturbance quantity connection.
5	TRK	IB	**TR**ac**K**ing. Input command for MANUAL mode.
6	KP	IB	**P**roportional coefficient.(1)
7	TINT	IT	**T**ime constant for **INT**egration.(1)
8	TDERIV	IT	**T**ime constant for **DERIV**ative action.(1)
9	TDLY	IT	**D**e**L**A**Y** **T**ime.(1)
10	BLKPOS	IB	Integrator **BL**oc**K**ed for **POS**IN>NEGIN.
11	BLKNEG	IB	Integrator **BL**oc**K**ed for POSIN<**NEG**IN.

Terminal Description (Continued)

No	Name	Type	Description
12	INTVAL	IR	**INT**egration **VAL**ue. Follow-up control of integrator.
13	LOLIM	IR	**LO**w **LIM**it value of corrective variable. Default value 0.
14	HILIM	IR	**HI**gh **LIM**it value of corrective variable. Default value 100.
15	RANGMIN	IR	Measurement **RANG**e **MIN**imum value.
16	RANGMAX	IR	Measurement **RANG**e **MAX**imum value.
20	O	OR	**O**utput.
21	DEV	OR	**DEV**iation output.
22	LIMACT	OB	Annunciation: **LIM**itation is **ACT**ive.
23	INTOUT	OR	**INT**egrator **OUT**put.
24	FPOSIN	OR	**F**eedback **POS**itive **IN**put.
25	FNEGIN	OR	**F**eedback **NEG**ative **IN**put.
26	FTRK	OR	**F**eedback **TR**ac**K**ing value.

(1) This value must be entered.

Function

Modes of operation

The element can be operated in either one of 2 operating modes:

- MANUAL mode (balancing mode)
- AUTOMATIC mode (controlled mode)

MANUAL mode

In MANUAL mode, a logic 1 signal is present at input TRK. In this case, controller output O is slaved without delay to the value of the balancing quantity present at input TRKVAL.

The D portion is ineffective in this operating mode.

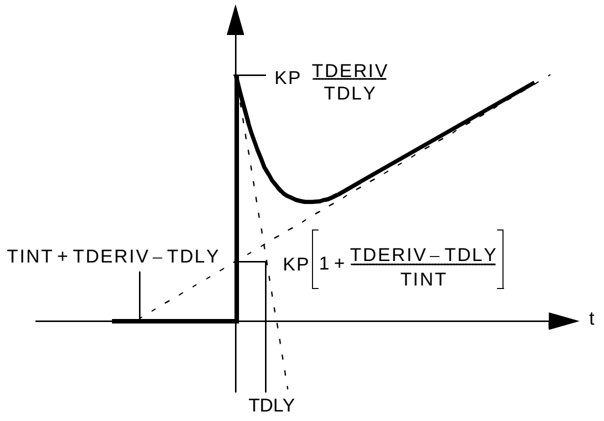
AUTOMATIC mode

In AUTOMATIC mode, a logic 0 signal is present at input TRK.

In this case, the transfer function of the PID controller is:

$$F(s) = KP * \frac{1+TINT*s}{TINT*s} * \frac{1+TDERIV*s}{1+TDLY*s}$$

The transfer function of the PID controller is represented in the following diagram:



The values for proportional coefficient, reset time, derivative action time, and delay time are defined by corresponding entries at inputs KP, TINT, TDERIV and TDLY.

In the case of TDERIV = TDLY, this controller behaves just like a PI controller.

Disturbance quantity connection

At input OFFSET, a disturbance quantity can be connected additionally. The time response of the controller is bypassed, and the disturbance quantity is directly switched through to output O.

Disturbance quantity limitation

Output variable O can be limited to an upper and a lower limit value using inputs HILIM and LOLIM.

When HILIM/LOLIM are effective, the integrator will be controlled accordingly by an internal function of the element, and the D portion will not be considered in that case.

Whenever upper and lower limit should overlap, the upper limit will be given priority.

When a limit value is effective, output O will follow this limit value without delay.

Integrator stop

The I portion of the controller can be inhibited at input BLKPOS by a logic 1 signal in the positive direction (POSIN > NEGIN), and at input BLKNEG by a logic 1 signal in the negative direction (POSIN < NEGIN).

Follow-up control of integrator

When the element is used as a master controller, and if a limit value is effective on a subordinate level, the integrator can be controlled smoothly via input INTVAL.

Input INTVAL can also be used for applying a synchronizing variable for several controllers operating in parallel.

Standardization of output signal

TO_NORM = 0

In this case, TDERIV, TDLY and TINT are related to an adjustment from measurement range RANGMIN to RANGMAX and not to the value 100.

TO_NORM = 1

In this case, input IN is standardized to RANGMIN, RANGMAX.

$$IN' = \frac{IN - RANGMIN}{RANGMAX - RANGMIN} * 100$$

where IN means POSIN and NEGIN.

In this case, POSIN has always be replaced by POSIN´ and NEGIN by NEGIN´ in this document.

Annunciation functions

Output DEV

The discrepancy between inputs NEGIN and POSIN will be put out at deviation output DEV (DEV = NEGIN – POSIN).

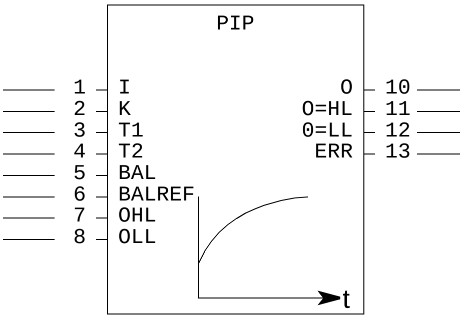
Output LIMACT

When the upper or lower limit responds, a logic 1 signal is put out at output LIMACT. To prevent repeated changing-over in the limit ranges, the output is given a fixed hysteresis of 0.5%.

PIP-Function PIP

Summary

PIP (**P**roportional **I**ntegrating **P**roportional function) is used to give a proportional effect and a limited integration effect. The output signal in the steady state is proportional to the input signal. The output signal can be limited with limit values specified at special inputs. The balancing function permits the output signal to follow an external reference and permits a bumpless return to the normal function. All transfers from static states are bumpless.



PC element PIP

Call PIP

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for actual value.
2	K	IR	Input for setting gain.
3	T2T1	ITR	**T**ime **1**. Input for time constant for lead.
4	T2	ITR	**T**ime **2**. Input for time constant for lag.
5	BAL	IB	**BAL**ance. Input for activation of following.
6	BALREF	IR	**BAL**ance **REF**erence. Input for reference value when following.
7	OHL	IR	**O**utput **H**igh **L**imit. Input for upper limit value.
8	OLL	IR	**O**utput **L**ow **L**imit. Input for lower limit value.
10	O	OR	**O**utput. Output signal.
11	O=HL	OB	**O**utput = **H**igh **L**imit. Output which is set to 1 if the output O is limited to the upper limit value.
12	O=LL	OB	**O**utput = **L**ow **L**imit. Output which is set to 1 if the output O is limited to the lower limit value.
13	ERR	OB	**ERR**or. Output set to 1 if OHL is less than OLL.

Function, Transfer Function

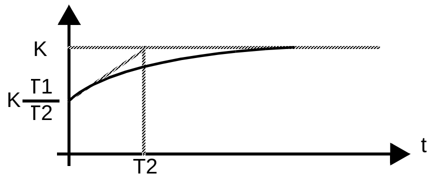
The step response in the time plane for the PIP function is:

$O(t) = I(t)\ (K+k1\times e^{-t/T2})$ where $k1 = K\ (T1/T2-1)$

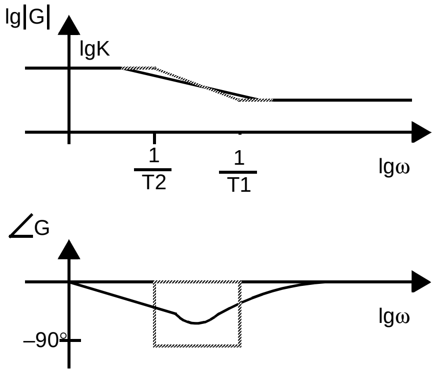
The transfer function for a PIP function is:

$G(s) = K(1+sT1)/(1+sT2)$ where $T2 > T1$

This has been implemented in the PIP element as a recursive function.



Step response



Bode diagram

Gain, Time Constants and Sampling Time

Certain constants are precalculated to make the execution time of the element as short as possible. The result is stored internally in the element. These constants are recalculated if T1, T2 or K are changed by more than 1/128 of their previous value or if the sampling time TS is changed. When recalculating a test is performed to check if T1 and T2 ≥ 2 × TS. If this is not the case, T1 or T2 respectively is set equal to 2 × TS.

Following

If BAL is set to 1, the regulator immediately goes into following and the output O is set to the value of the input BALREF. If the value at BALREF exceeds the output signal limits, the output is set to the limit value concerned. Return to dynamic state is bumpless.

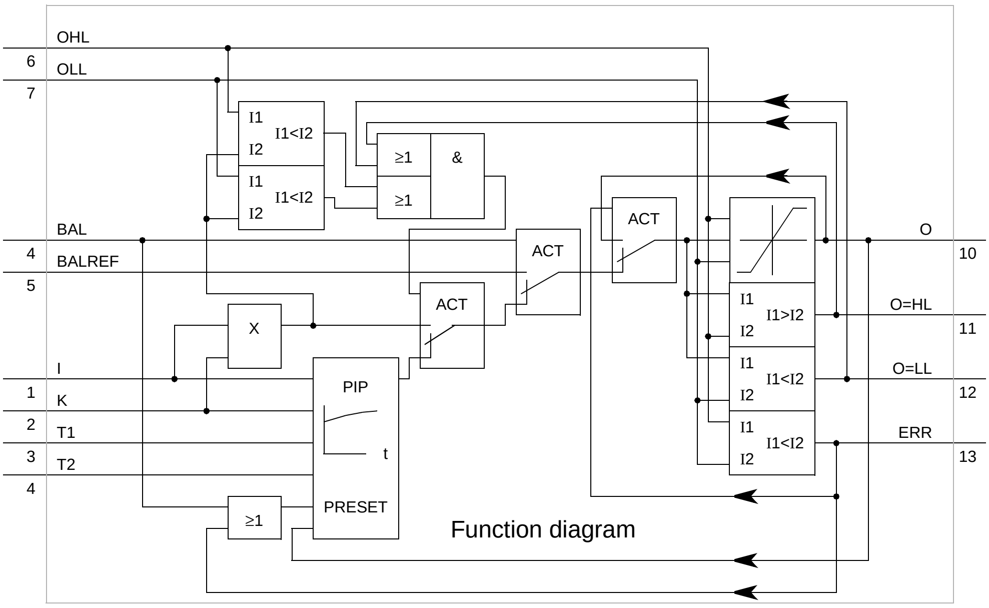
Limitation Function

The limitation functions limits the output signal to the values at the inputs OHL for the upper limit and OLL for the lower limit. If the actual value exceeds the upper limit, the output O=HL is set to 1 and if it falls below the lower limit, the output O=LL is set to 1.

When the limitation status has been detected, a check is made each time the element is executed to determine whether K × I(t) exceeds the output signal limitations. If such is the case, the limitation status remains. If not, the calculation of the output signal is performed by the algorithm in the normal way.

Normal updating of the internal status is sufficient to ensure bumpless return to the dynamic status. The element checks that the upper limit value OHL is greater than the lower limit value OLL. If this is not the case, the output ERR is set to 1.

While the error status persists, the outputs O=HL, O=LL and O retain the values they had in the sample before that in which the error occurred. After an error the return to a dynamic state is bumpless, in the same way as in the case of following.

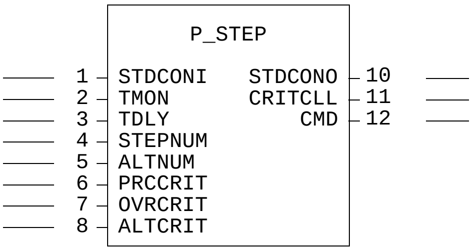


Function diagram

“P” Step P_STEP

Summary

The element P_STEP (**“P” STEP** function) is used within a sequential control to give the commands step by step for running (ON program) or stopping (OFF program) of a functional group. One element corresponds to one step. The same elements will be used for the ON program as well as for the OFF program.



PC element P_STEP

Call P_STEP

Terminal Description

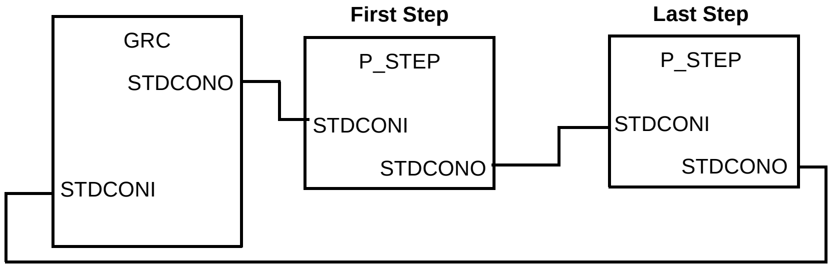
No	Name	Type	Description
1	STDCONI	IG	**ST**andar**D** **CON**nection **I**n. Connection from a P_STEP or a GRC element.
2	TMON	IT	**MON**itoring **T**ime.
3	TDLY	IT	**D**e**L**a**Y** **T**ime.
4	STEPNUM	II	**STEP NUM**ber
5	ALTNUM	II	**ALT**ernative step **NUM**ber.
6	PRCCRIT	IB	**PR**o**C**eeding **CRIT**eria.
7	OVRCRIT	IB	**OV**e**R**running **CRIT**eria.
8	ALTCRIT	IB	**ALT**ernative proceeding **CRIT**eria.
10	STDCONO	OG	**ST**andar**D** **CON**nection **O**ut. Connection to the next P_STEP or a GRC element.
11	CRITCLL	OB	**CRIT**eria **C**a**LL** for indication of the criteria not being fulfilled. (This output is active too when input RELCMD of the element GRC is: RELCMD = 0.)
12	CMD	OB	Step **C**o**M**man**D**.

Note

- If TMON = 0, the corresponding step will not be monitored.
- The first step of the ON program must have assigned step number 1.
- Within a program, there must be a consecutive numbering of the steps.

Connections of the Element GRC and P_STEPs

Data transfer between the element GRC and the elements P_STEP will be done via the connection STDCONI–STDCONO. All steps and the GRC have to be included in this chain.



Function

Step Number

Each step in a chain of steps must be identified with an unique ascending step number. The first step of the ON program must have assigned step number 1. On the basis of the value in the STDCONI – STDCONO telegram given by the GRC element, the step decides if he will be active or if the information in the telegram will be forwarded to the next step in the chain.

Step Command Output

One cycle after actuation of the step, the step command at output CMD will be carried out if a logic 1 signal is present at input RELCMD of the element GRC. (Release for commands)

Continuous Running of the Program by Proceeding Criteria

The process criterion (proceeding condition) will be connected to input PRCCRIT. When the proceeding condition is given, there will be forwarded to the next step under consideration of an eventually defined delay time TDLY. (Refer to “Delay time”.)

Overrunning Criteria

For synchronizing of the control with process state (search program), it will be necessary to overrun certain steps without a given command. For that purpose, input OVRCRIT will be used. A logic 1 signal at this input immediately effects forwarding to the next step without a given command. An eventually defined delay time TDLY will be ignored.

Alternative Proceeding Criteria

Input ALTCRIT is used to realize jump programs. A logic 1 signal at this input effects jumping to the step which is specified at input ALTNUM after an eventually defined delay time TDLY has expired.

Time monitoring

The time for executing each command can be monitored by the time which is specified at input TMON. The permissible time can be adjusted for each step. The supervision function of the time will be done by the element GRC as well as the generation of a respective annunciation of an alarm when the time has been exceeded. If the monitoring time of a step shall be put out of use, a logic 0 signal at input TMON must be present.

Delay time

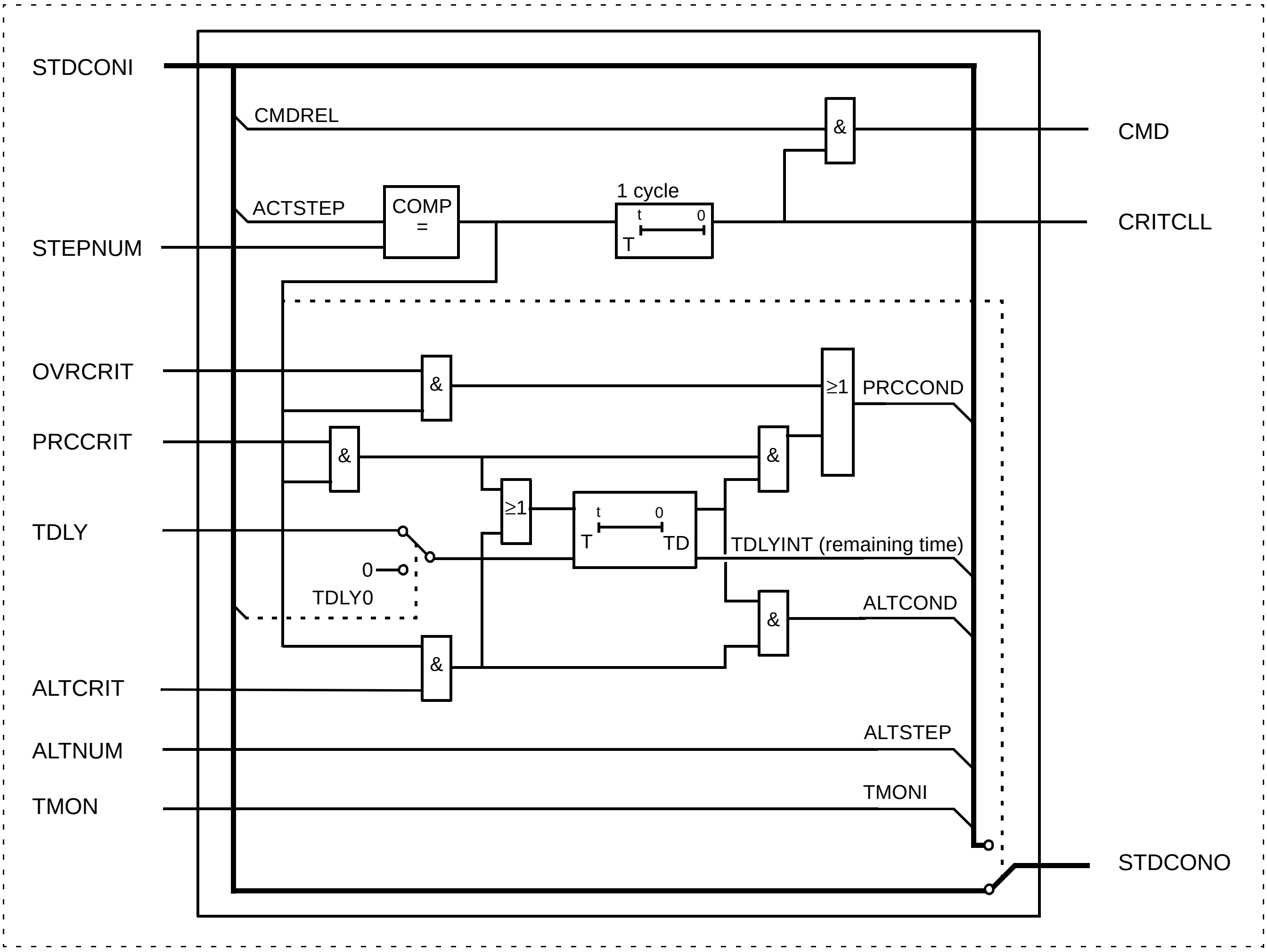
A delay time can be defined at input TDLY. With actuated step and given proceeding criteria or alternative proceeding criteria, the step will delay output of the respective signal to the element GRC according to the delay time which is specified at input TDLY. Additionally after a criteria was given, the step computes the remaining delay time and gives this message to the GRC.

Criteria Call

Output CRITCLL is used to indicate criteria not yet being fulfilled. One cycle after actuation of the step, this output will carry out a logic 1signal. The output CRITCLL will be active too when input RELCMD of the element GRC is RELCMD = 0.

Indication of a Step

The active (set) step will be indicated at output ACTSTEP of the element GRC.

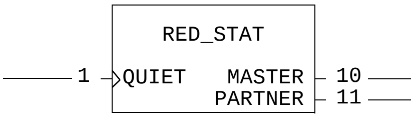


Function diagram

Redundancy Status RED_STAT

Summary

RED_STAT is used in a redundancy configuration to make system redundancy status of Processor Modules PM645 and PM646 available for the application program and to allow an external acknowledge in case of a defective slave.



PC Element RED_STAT

Call RED_STAT

Terminal Description

No	Name	Type	Description
1	QUIET	IB	For external acknowledge of a defective slave.
10	MASTER	OB	Redundancy status of Processor Module: 1 = master; 0 = stand by.
11	PARTNER	OB	Redundancy status of partner Processor Module: 1 = partner ok; 0 = partner disturbed.

Function

The input QUIET is used to acknowledge a disturbance on a defective redundancy slave Processor Module with an external signal. If the disturbance is still pending on the Processor Module an acknowledge is not possible.

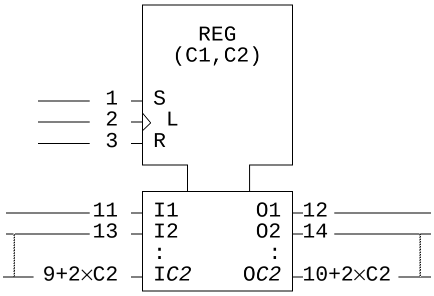
A signal “1” on the output MASTER indicates that the Processor Module is running as master in this redundant configuration. A “0” at this output signals the stand by mode of the Processor Module.

The output PARTNER gives information about the partner Processor Module in a redundant configuration. “1” indicates that the partner is running faultless. A “0” on the output signals that an error is still pending or an old error message was not acknowledged.

Register REG

Summary

Register REG (**REG**ister) is used as a memory element with an optional number of positions. The value of the data can be integer, real number, Boolean or time.



PC element REG

Call REG (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of places	1…35

Terminal Description

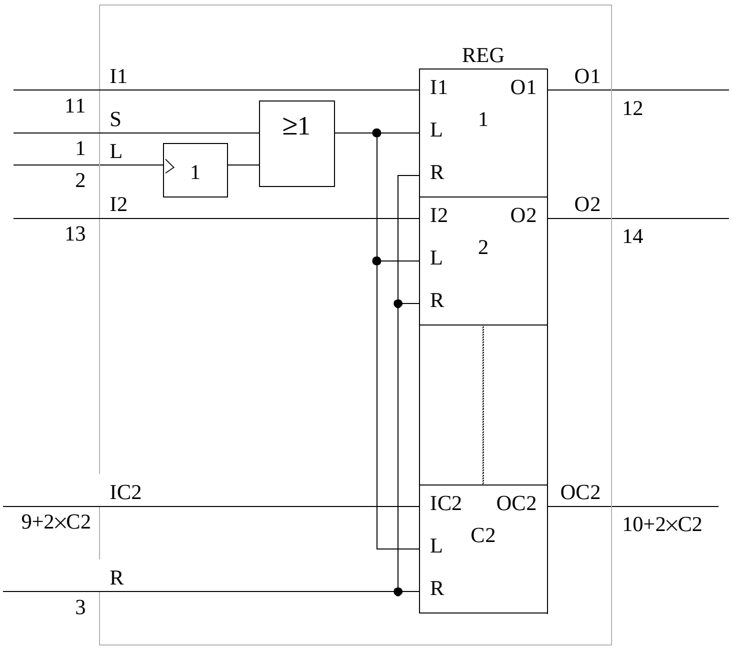
No	Name	Type	Description
1	S	IB	Set. Input for loading new data, each time the element is executed. When S is set to 0, the last data loaded remains.
2	L	IB	**L**oad. Dynamic input for loading data to the register. Loading is performed when L goes to 1.
3	R	IB	**R**eset. Input for clearing the register. R prevents all further entry while it is 1.
11	I1	I*C1*	**I**nput **1**. Input data to position 1.
12	O1	O*C1*	**O**utput **1**. Output data from position 1.
13	I2	I*C1*	**I**nput **2**. Input data to position 2.
14	O2	O*C1*	**O**utput **1**. Output data from position 2.
: 9+2×C2	I*C2*	I*C1*	**I**nput ***C2***. Input data to position C2.
10+2×C2	O*C2*	O*C1*	**O**utput ***C2***. Output data from position C2.

Function

When input L becomes 1, the register is loaded with data from inputs I1… I*C2*. Data previously in the register is replaced. If input S is set, loading is performed as above each time the register is executed. When S is reset after having been set, the data most recently loaded remains until the element is executed again, with input S, L or R set. Input S overrides L so that when input S is set, L has no effect.

Clearing

When input R is set, the register is cleared and all further entry is prevented. R overrides both S and L.

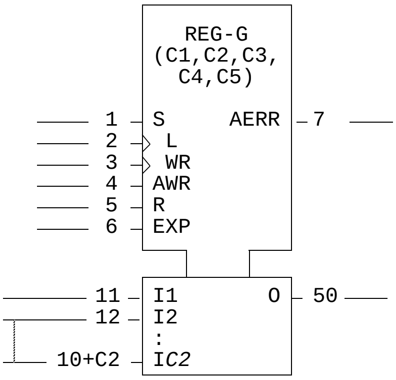


Function diagram

Group Data Register REG-G

Summary

REG-G (**REG**ister **- Gr**oup data) is used to assemble individual variables to a single variable of group type. The data type can be integer, real number, Boolean or time.



PC element REG-G

Call REG-G (C1, C2, C3, C4, C5)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of inputs	0…30
C3	Selection of expander input EXP	0…1
C4	Number of places in group data for expander input EXP	0…254
C5	Number of places in group data for the output O.	0…255
C2 + C3 × C4 = C5		

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et. Input for loading new data each time the element is executed. The latest loaded data remains when S is reset.
2	L	IB	**L**oad. Dynamic input for loading data to the register. Loading is performed when L goes to 1.
3	WR	IB	**WR**ite. Dynamic input for changing of data at the place specified by AWR.
4	AWR	II	**A**ddress **WR**ite. Input for address to the data to be changed according to WR.
5	R	IB	**R**eset. Input for clearing the register. R prevents all further entry while at 1.
6	EXP	IG*C4C1*	**EXP**ander input. Input for group data to be linked with data at inputs I1…I*C2*.

Terminal Description (Continued)

No	Name	Type	Description
7	AERR	OB	**A**ddress **ERR**or. Output which is set (to 1) if the address at AWR is greater than the number of inputs or is negative.
11	I1	I*C1*	**I**nput **1**. Input data to position 1.
12	I2	I*C1*	**I**nput **2**. Input data to position 2.
: 10+C2	I*C2*	I*C1*	**I**nput ***C2***. Input data to position C2.
50	O	OG*C5C1*	**O**utput. Output for group data.

Function

REG-G combines individual variables of the same data type to a single group variable. The element has an expander input EXP for a group variable which can be used when more than 30 variables are to be linked together. Input S overrides L so that when input S is set, L has no effect.

Normal Assembly

When input S is set, data is continuously assembled at the group variable of the output. The group variable of the output consists firstly of group data from the input EXP and then of the values from the inputs I1…I*C2*. The element acts as a latch when the input S is reset, the latest data assembled then remaining at the output.

Loading

If input L goes from 0 to 1, an assembly is performed to output O during this program cycle in the same way as during normal assembly. If S or R is set, L has no effect.

Changing of Data at an Optional Place

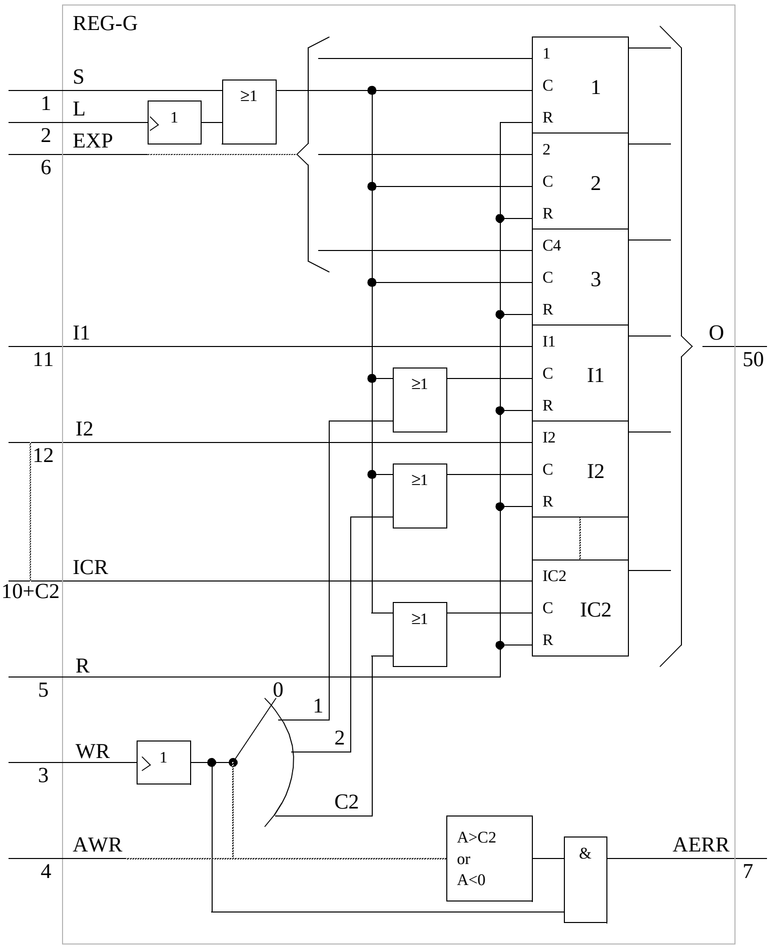
One of the ordinary data inputs I1…I*C2* is used for changing of data at a special place in REG-G. The address (integer 1…C2) of the place at which data is to be changed is stated at the input AWR. The new data value is entered from the input to the relevant place at the moment that the input WR goes to 1. If AWR = 0 and the input WR goes to 1, group data is read from the input EXP to their different places. Places which correspond to ordinary inputs are not affected.

Clearing

When input R is set, data at all places in the group register are cleared and all further entry is prevented. R overrides both S and L.

Supervision

The address at the input AWR is checked and if its value is greater than the number of inputs or negative, the error signal output AERR is set. When an error is detected AERR is set during one cycle if input WR is set. No place in the register is affected when an error occurs.

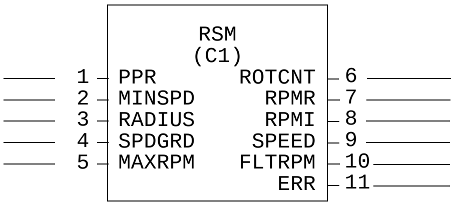


Function diagram

Rotational Speed Measurement RSM

Summary

The element RSM (**R**otational **S**peed **M**easurement) is used for measuring the rotational speed with identification of the direction of the rotation.



PC element RSM

Call RSM (C1)

Parameter	Significance	Permissible values
C1	Selection of one of the both possible measuring inputs 1 or 2 being used.	0…1

Terminal Description

No	Name	Type	Description
1	PPR	II	**P**ulses **P**er **R**otation. This value must be entered.
2	MINSPD	IT	Time between two pulses at **MIN**imal rotational **SP**ee**D**. When the time between two pulses is longer than MINSPD, outputs RPMR, RPMI and SPEED are set to zero.
3	RADIUS	IR	**RADIUS** of the gear-wheel.
4	SPDGRD	IR	**SP**ee**D** **GR**a**D**ient. Maximum permitted changes of RPMR between 2 calls of the element. With that, faults of rotational speed will additionally be supervised. During Fault, the terminal FLTRPM will be set.
5	MAXRPM	IR	**MAX**imum **P**er**M**itted **R**otational speed. When MAXRPM is exceeded, FLTRPM signal is put out for protection. If not used leave it open.
6	ROTCNT	OIL	**ROT**ation **C**ou**NT**er. Number of measured teeth. When starting, this input can have any offset value.
7	RPMR	OR	**R**otations **P**er **M**inute rpm, type **R**eal. RPMR = (number of measured teeth/PPR).
8	RPMI	OIL	**R**otations **P**er **M**inute rpm, type **I**nteger long.
9	SPEED	OR	Rotational **SPEED**. SPEED = 2×π×RPMR×RADIUS.
10	FLTRPM	OB	**F**au**LT** signal if the maximum rotational speed MAX**RPM** is exceeded. This signal is not stored, but updated after each execution cycle of the PC element. To switch off the processor module’s relays, connect FLTRPM to terminal REL_OFF of the processor module DB element.
11	ERR	OB	**ERR**or output.

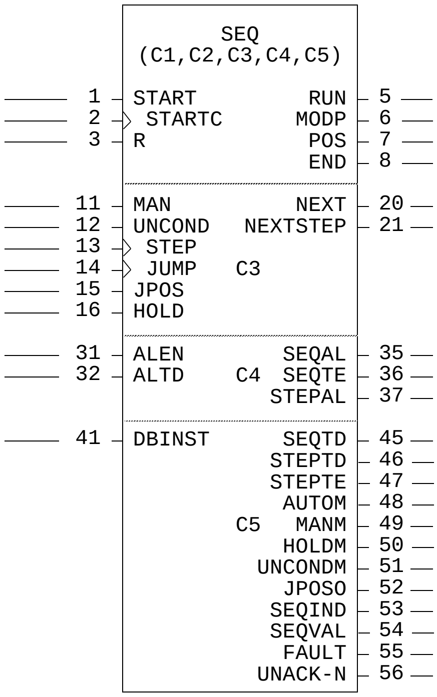
Note

The RSM element must at least be called every 20 seconds. For each measuring channel the RSM must not be used more than once.

Sequence Header SEQ

Summary

SEQ is used to control a sequence.



PC element SEQ

Call SEQ (C1, C2, C3, C4, C5)

Parameter	Significance	Permissible values
C1	Cyclicity in ms.	Allowed values are between 2 ms and 20,000 ms in steps of 2 ms.
C2	Place in the cycle time table and/or scheduling strategy.	1…251: Place in cycle time table (cyclic activation). 252: Activation at initialization time after power down. 255: Event driven by onboard I/O.
C3	Selection of manual control	0…1
C4	Selection of time function	0…1
C5(1)	Selection of Operator Station function	Same values allowed as in Advant Controller 400 Series and MP 200/1. **No effect** in Advant Controller 100 Series.

(1) Parameter C5 is allowed for compatibility reasons to the Advant Controller 400 Series or MP 200/1-system. You can exchange AMPL-source code between the 400 Series or MP 200/1 and the 100 Series without problems.

Terminal Description

No	Name	Type	Description
1	START	IB	Input for start of sequence.
2	STARTC	IB	**START C**lock. Dynamic input for start of the sequence. Used when the sequence is to be run through once only.
3	R	IB	**R**eset. Clearing of the elements of the sequence.
5	RUN	OB	Indicates that the sequence is in operation.
6	MODP	OB	**MOD**ify **P**ermission. Specifies if the sequence may be modified from the engineering station.
7	POS	OI	**POS**ition. Output which specifies the step number of the active step.
8	END	OB	Output which is active when the last step of the sequence is active.
If C3=1, the following terminals will be shown:			
11	MAN	IB	**MAN**ual. Input which is to be set for manual stepping of the sequence.
12	UNCOND	IB	**UNCOND**itional. Input which permits stepping even when the condition for the next step is not satisfied.
13	STEP	IB	Input for step change.
14	JUMP	IB	Input for jump to step specified by JPOS.
15	JPOS	II	**J**ump **POS**ition. Input which specifies the last part of the record designation for the step to which the jump is to be made.
16	HOLD	IB	Input which is used to stop the execution.
20	NEXT	OB	Output which is set when any condition for step change or jump is satisfied or when UNCOND is set.
21	NEXTSTEP	OI	Step to which stepping in manual mode is performed. Is connected to the corresponding input of the data base element when MMC function is used.
If C4=1, the following terminals will be shown:			
31	ALEN	IB	**AL**arm **EN**able. Input for start and stop of time measurement.
32	ALTD	IT	**AL**arm **T**ime **D**elay. Input for time after which SEQAL is to be set.
35	SEQAL	OB	**SEQ**uence **AL**arm. Output which is set when the time specified on ALTD has elapsed.
36	SEQTE	OT	**SEQ**uence **T**ime **E**lapsed. Output for elapsed time.
37	STEPAL	OB	**STEP AL**arm. Output for summary signal which indicates that the time set in an active subordinate step has elapsed.
If C5=1, the following terminals will be shown:			
41	DBINST	II	**D**ata **B**ase **INST**ance. The instance number of the sequence in the data base index, alternatively the name of the instance.
45	SEQTD	OT	**SEQ**uence **T**ime **D**elay. Output that indicates ALTD.
46	STEPTD	OT	**STEP T**ime **D**elay. Output giving ALTD for the active step.

Terminal Description

No	Name	Type	Description
47	STEPTE	OT	**STEP T**ime **E**lapsed. Output for time that has elapsed in the timer of the active step.
48	AUTOM	OB	**AUTO M**ode. Output which is set when operating in mode AUTO.
49	MANM	OB	**MAN**ual **M**ode. Output which is set when operating in mode MAN.
50	HOLDM	OB	**HOLD M**ode. Output which is set when operating in mode MAN.
51	UNCONDM	OB	**UNCOND**itional **M**ode. Output which is set when operating in mode MAN with UNCOND.
52	JPOSO	OI	**J**ump **POS**ition **O**utput. Output that indicates the jump target JPOS.
53	SEQIND	OI	**SEQ**uence **IND**ications. Output for status information to the data base element.
54	SEQVAL	OA20	**SEQ**uence **VAL**ues. Output for value information to the data base element.
55	FAULT	OI	Output for error information to the data base element.
56	UNACK-N	OB	**UNACK**nowledged**-N**. Output to the data base element for clearing of dialogue. The dialogue is cleared with each execution.

Function

The sequence header SEQ is a superior execution controlling element for a number of sequence steps. How often the sequence is to be executed is specified with the call parameter C1 and the place in the cycle time table on which the order of execution of the execution units with the same cycle time is based is specified with the call parameter C2. The basic function can be extended with the call parameters C3, C4 and C5. Manual control of the sequence can be obtained by setting C3 to 1 at call. A time function can be obtained by setting C4 to 1.

Start of Sequence

With normal execution, the sequence can be started with a pulse at the input START. The pulse is registered in an internal memory in the sequence header. If a fixed 1 is connected to START, the sequence can be connected for automatic restart after resetting. If the sequence is to be run through only once, it must be started with STARTC. STARTC is a dynamic input and the internal memory of the sequence header is only affected when the input status is changed from 0 to 1. After a start, the output RUN is set and the condition input for the first step is checked. If the condition is satisfied, the first step and its elements are executed.

Clearing

If the input R is set, the sequence is executed in the reset mode. This means that all outputs on the elements within all of the steps of the sequence are given initial values, which in most cases is the 0-value of the data. Reading to the data base is done with the data value 0 for all data types. For the steps, reset execution is performed only during the cycle when R goes to 1. For the sequence header, reset execution is performed during each cycle while R is set to 1. Input R overrides the inputs START, STARTC and HOLD.

RUN

The output RUN is set when the sequence is under execution, that is when started with START or STARTC. RUN is cleared by R, HOLD and blocking functions introduced from service aids.

MODP

MODP is always true.

POS and END

The output POS specifies, with an integer (I), the last part of the record designation of the sequence step which is active. If POS is 0, the sequence header is active. The output END is set when the last step in the sequence is active.

Manual Execution

For manual stepping of a sequence, the input MAN must be set to 1. The sequence is started in the same way as with automatic stepping by setting START or STARTC to 1. When the condition for step change, or a jump condition, is satisfied, the output NEXT is set, and the step change can be performed with a pulse at the input STEP. Jumps have priority to ordinary step change.If stepping is required without the condition for the next step being satisfied, the input UNCOND must be set to 1 when STEP is set to 1.

Jump in Sequence

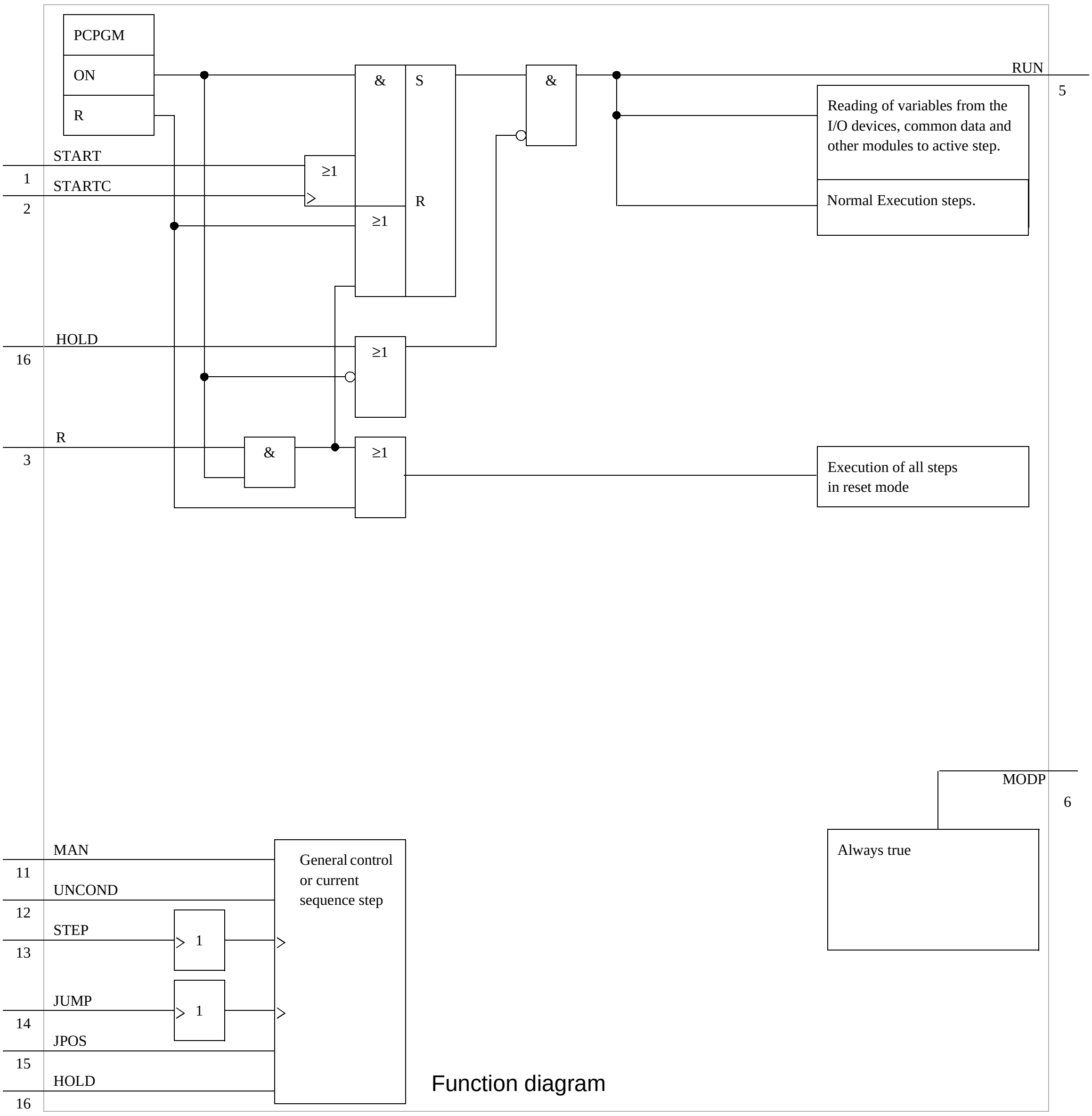
A jump can be obtained by setting the input JUMP. The jump is made to the step which is specified at the input JPOS. The step is specified by the last part of its record designation. The jump is performed irrespective of the status at the inputs MAN, STEP and UNCOND. If the MMC function is used and a jump is to be initiated by the operator via the dialogue, the operation mode must be MANM. A jump is not possible if HOLD or R is set.

HOLD

When HOLD is set, execution of active sequence steps is stopped. The calculated data remain until the next time the sequence step is executed. The execution is stopped and the output RUN is reset while HOLD is set to 1. The RUN output at the step header remains set. HOLD also stops the timers in the sequence header and the step headers.

Time Functions

If the call parameter C4 is given as 1, a time function associated with the sequence header is obtained. A common signal STEPAL is obtained which indicates any time function which has elapsed in any of the steps. The switch on of the timer in the sequence header is delayed with accumulating time measurement controlled by the input ALEN. If ALEN is reset, the timer is stopped but its contents remain and when ALEN is set, the time measurement restarts from the value that applied when ALEN was reset. The setting of HOLD has the same effect on the timer as resetting ALEN. The timer can only be started when the sequence is active. When the time specified at the input ALTD has elapsed, the output SEQAL is set. When ALTD is set equal to 0, SEQAL is set to 1 irrespective of the state at input ALEN. The output SEQTE gives the time which has elapsed with the timer active. The timer is cleared when the complete sequence is cleared with the input R. A timer corresponding to that in the sequence header can be selected for the sequence steps. The output STEPAL, which specifies if the time set in the timer for the active step has elapsed, is located in the sequence header. STEPAL is set to 1 if the delay specified for the step concerned has elapsed.

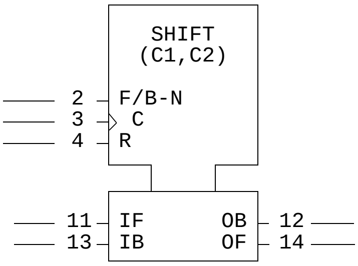

PCPGM
ON
R
START
1
STARTC
2
≥1
&
S
R
≥1
&
RUN
5
Reading of variables from the I/O devices, common data and other modules to active step.
Normal Execution steps.
HOLD
16
≥1
R
3
&
≥1
Execution of all steps in reset mode
MODP
6
Always true
MAN
11
UNCOND
12
STEP
13
1
JUMP
14
1
JPOS
15
HOLD
16
General control or current sequence step

Function diagram

Shift Register SHIFT

Summary

Shift register SHIFT is used as a shift register of up to 250 positions. Data can be integers, real numbers, Boolean or time.



PC element SHIFT

Call SHIFT (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of places	2…250

Terminal Description

No	Name	Type	Description
2	F/B-N	IB	**F**orward/**B**ackward-**N**. Input for selection of shift direction. Shifting is forward when F/B-N = 1 and backward when F/B-N = 0.
3	C	IB	**C**ontrol. Dynamic input for shifting.
4	R	IB	**R**eset. Input for clearing the shift register. All further shifting is prevented when R = 1.
11	IF	I*C1*	**I**nput **F**orward. Input data with forward shift.
12	OB	O*C1*	**O**utput **B**ackward. Output data with backward shift.
13	IB	I*C1*	**I**nput **B**ackward. Input data with backward shift.
14	OF	O*C1*	Output Forward. Output data with forward shift.

Function, Forward Shift

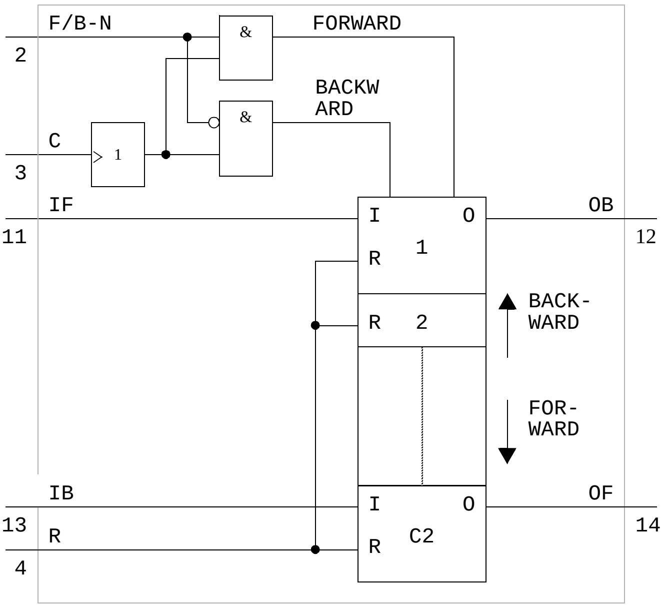
When input C goes to 1 and the input F/B-N is 1, data is shifted forwards in the register and data is read into position 1 from input IF. Data which before the shifting was located at output OF, is replaced at the shift forward by data at the position next to last.

Backward Shift

When input C goes to 1 and the input F/B-N is 0, data is shifted backwards in the register and data is read into position C2 from input IB. Data which before the shifting was located at output OB, is replaced at the shift backward by data from position 2.

Clearing

The input R clears the shift register and prevents all further reading into the shift register.



Function diagram

Slave Header SLAVEM

Summary

SLAVEM (**SLAVE M**odule) is used for execution control of a slave module.

`SLAVEM(C1,C2)`

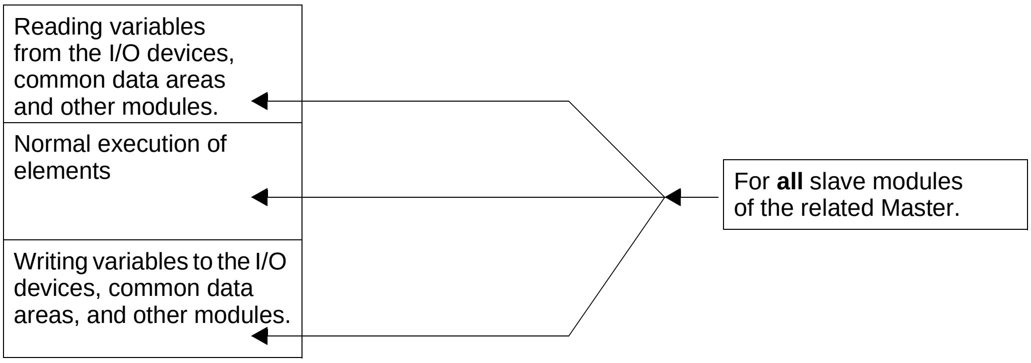
PC element SLAVEM

Call SLAVEM (C1, C2)

Parameter	Significance	Permissible values
C1	Identity of execution controlling master header	The second part of the PC record designation of the master header.
C2	Currently not used	–

Function

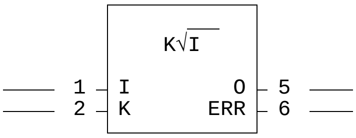
The slave header SLAVEM is an element header for a slave module. The numerical identity of the master header associated with the slave module is specified with the call parameter C1. This then becomes the second part of the PC record designation of the master as this must always be at the level directly subordinate to a program header. How often and in what order the slave module is to be executed in relation to other execution units is determined by the superior master header. The slave modules within the same master are executed in the order of the documentation. Blocking from service aids and PC programs is determined entirely by the master.


Reading variables from the I/O devices, common data areas and other modules.
Normal execution of elements
Writing variables to the I/O devices, common data areas, and other modules.
For **all** slave modules of the related Master.

Square Root SQRT

Summary

SQRT (**SQ**uare **R**oo**T**) is used to calculate the square root of a real number. The result can be multiplied by an optional value.



PC element SQRT

Call SQRT

Terminal Description

No	Name	Type	Description
1	I	IR	**I**nput. Input for number of which the square root is to be calculated.
2	K	IR	Input for multiplication factor K.
5	O	OR	**O**utput. Output for the product of the value at input K and the square root of the value at input I.
6	ERR	OB	**ERR**or. Output which is set if the value at input is negative.

Function

The square root of the value at input I is calculated. The result is multiplied by the value at input K. The product is stored at the output O.

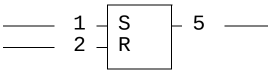
Supervision

If the value at the input I is negative, the error output ERR is set to 1. The value 0.0 is then stored at the output O.

Set/Reset Memory SR

Summary

The memory element SR (**S**et **R**eset memory) is used as a memory for Boolean variables.



PC element SR

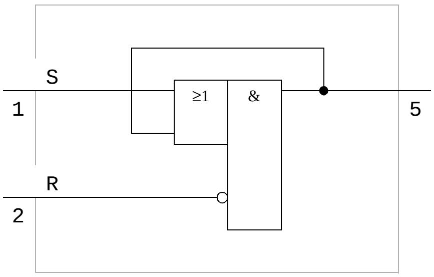
Call SR

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et input.
2	R	IB	**R**eset input which overrides the set input.
5	–	OB	Output from the memory element.

Function

The element output is set (to 1) if the set input is set at the same time that the reset input is reset (to 0). If the reset input is set, the output is unconditionally reset.

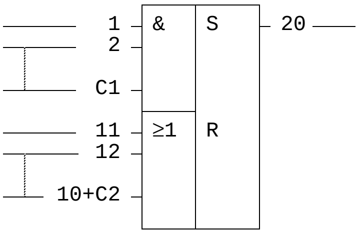


Function diagram

Memory Elements SR-AA, SR-AO, SR-OO, SR-OA

Summary

SR-AA, SR-AO, SR-OA and SR-OO (**S**et **R**eset memory - **A** = And, **O** = Or) are used as a memory for Boolean variables. The elements have different gate elements for condition generation at the set and reset inputs.



PC element SR-AO

Call SR-AA (C1, C2), SR-AO (C1, C2), SR-OA (C1, C2) SR-OO (C1, C2)

Parameter	Significance	Permissible values
C1	Number of inputs in the set gate	1…9
C2	Number of inputs in the reset gate	1…9

Terminal Description

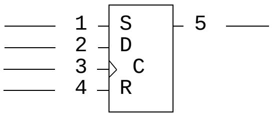
No	Name	Type	Description
1	–	IB	Input to the set function.
2	–	IB	Input to the set function.
: *C1*	–	IB	Input to the set function.
11	–	IB	Input to the reset function.
12	–	IB	Input to the reset function.
: 10+*C2*	–	IB	Input to the reset function.
20	–	OB	Memory element output.

Set/Reset Memory with Data Input SR-D

Summary

The memory element SR-D (**S**et **R**eset memory - **D**ata input)

Call SR-D



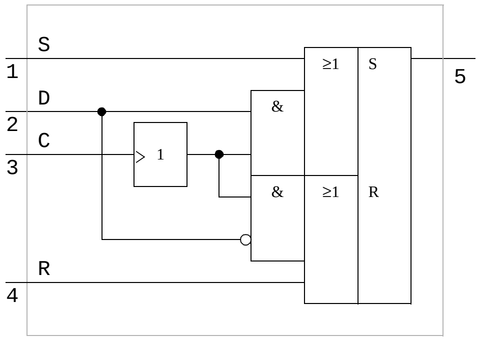
PC element SR-D

Terminal Description

No	Name	Type	Description
1	S	IB	**S**et input.
2	D	IB	**D**ata input.
3	C	IB	**C**lock. Dynamic input for entry of data from the D input.
4	R	IB	**R**eset input which overrides all other inputs.
5	–	OB	Output from the memory element.

Function

If only the S and R inputs are used, SR-D functions as an ordinary SR element. When the input R is reset and the input C goes to 1, the value at the input D is stored at the output 5. When the input R is set, the output 5 is unconditionally reset, that is R overrides the other inputs.

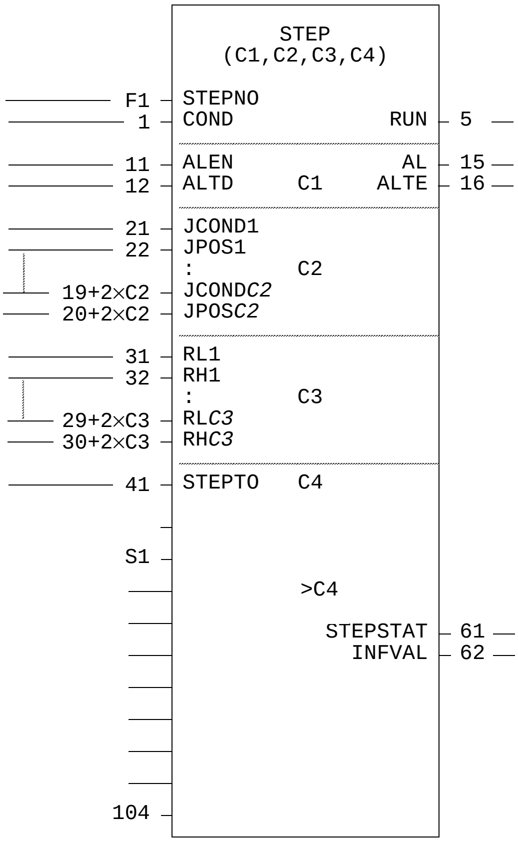


Function diagram

Step Header STEP

Summary

The step header STEP is used to partition a sequence into sequence steps.



PC Element STEP

Call STEP (C1, C2, C3, C4, C5, C6, C7, C8)

Parameter	Significance	Permissible values
C1	Selection of time function	0…1
C2§	Number of jumps	0…4
C3	Number of reset groups	0…4
C4	Selection of alternative execution order	0…1
>C4(1)	Same values allowed as in Advant Controller 400 Series or MP 200/1 but in Advant Controller 100-system: **no effect.**	

(1) Parameters >C4 is allowed for compatibility reasons to the 400 Series or MP 200/1-system. So you can exchange AMPL source code between Advant Controller 400 Series or MP 200/1 and Advant Controller 100 Series without problems.

Terminal Description

No	Name	Type	Description
F1	STEPNO	II	Parameter input for **STEP N**umber. F = 0 for automatic step numbering according to the last number of the item designation. F = 1…999 for manual step numbering.
1	COND	IB	**COND**ition. Condition input for stepping.
5	RUN	OB	Indicates that the step is active.
If C1=1, the following terminals will be shown:			
11	ALEN	IB	**AL**arm **EN**able. Input for start and stop of time measurement.
12	ALTD	IT	**AL**arm **T**ime **D**elay. Input for alarm time.
15	AL	OB	**AL**arm. Output for alarm. Set (to 1) when the time specified at the input ALTD has elapsed.
16	ALTE	OT	**AL**arm **T**ime **E**lapsed. Output for elapsed time.
If C2=1, the following terminals will be shown:			
21	JCOND1	IB	**J**ump **COND**ition **1**. Input for jump to step specified by JPOS1.
22	JPOS1	II	**J**ump **POS**ition **1**. Specifies which step the jump is made to if JCOND1 is set.
23	JCOND2	IB	**J**ump **COND**ition **2**. Input for jump to step specified by JPOS2.
24	JPOS2	II	**J**ump **POS**ition **2**. Specifies which step the jump is made to if JCOND2 is set.
: 19+2×C2	JCOND*C2*	IB	**J**ump **COND**ition ***C2***. Input for jump to step specified by JPOS*C2*.
20+2×C2	JPOS*C2*	II	**J**ump **POS**ition ***C2***. Specifies which step the jump is made to if JCOND*C2* is set.
If C3=1, the following terminals will be shown:			
31	RL1	II	**R**eset **L**ow limit **1**. Input for the number of the first step in the first group of steps to be cleared.
32	RH1	II	**R**eset **H**igh limit **1**. Input for the number of the last step in the first group of steps to be cleared.
33	RL2	II	**R**eset **L**ow limit **2**. Input for the number of the first step in the second group of steps to be cleared.
34	RH2	II	**R**eset **H**igh limit **2**. Input for the number of the last step in the second group of steps to be cleared.
: 29+2×C3	RL*C3*	II	**R**eset **L**ow limit ***C3***. Input for the number of the first step in the last group of steps to be cleared.
30+2×C3	RH*C3*	II	**R**eset **H**igh limit ***C3***. Input for the number of the last step in the last group of steps to be cleared.
If C4=1, the following terminals will be shown:			
41	STEPTO	II	**STEP T**O. This input modifies the execution order of the sequence. The input determines which step should be next step in the sequence. To be used in convergence of a sequence (see the manual ”MasterPiece - PC programming”).

Function

The step header STEP is a supervisory element header for partitioning a sequence into sequence steps. Elements, function modules and blocks can be directly subordinate to a step header. How often and in which order the step is to be executed in relation to other execution units is determined in the supervisory sequence header.

The element has a basic function which may be extended by setting one or more of the call parameters C1, C4 to a value different from 0. A time function can be obtained with the call parameter C1. Jump and reset functions can be obtained with C2 and C3. With C4 an alternative execution order is obtained.

Step Number

If parameter input STEPNO has the default value 0, step numbering is performed automatically according to the last number of the STEP element's item designation. If you wish to have other step numbers, number 1…999 can be given explicitly at the STEPNO parameter input. It is the user's responsibility to give each step an unique step number.

Basic Version

The basic version of STEP includes only the condition input COND for stepping and the output RUN which indicates that the step is active.

Normal Execution

A step change takes place if the sequence is executing normally in accordance with the conditions in the sequence header and the COND input of the following step is set to 1. A step may also become active through a jump from the sequence header.
A step may become active without COND being set to 1. This occurs when UNCONDitional on the sequence header is used.
With the step change, the RUN output of the active step is reset, after which the elements in this step are executed once in the normal way. The RUN output of the new active step is then set and the elements in this step are then executed. If the input HOLD on the supervisory sequence header is set to 1, execution of the active step is stopped and the RUN output remains set.

Time Function

If the call parameter C1 is specified as 1, a timer with accumulating time measurement, controlled by the input ALEN,
is obtained. If ALEN is reset, the timer is stopped but the contents remain and when ALEN is set, the time measurement continues from the value which applied when ALEN was reset. Setting HOLD in the supervisory sequence header gives the same result on the time measurement circuit as the resetting of ALEN. The timer can only be started when the step is active. Elapsed time is specified at the output ALTE. When the time specified at the input ALTD has elapsed, the output AL is set. The timer is reset when the step is cleared by another step (see below) or by the setting of the R input in the supervisory sequence header.

Jump

When any of the condition inputs JCOND1…JCOND*C2* are set, a jump is performed to the step specified at the corresponding input of JPOS1…JPOS*C2*. If several condition inputs are set at the same time, a jump is made to the step specified by the input with the lowest sequential number. If the conditions for step change are satisfied at the same time as those for a jump, the jump is performed. No jumps can be performed when the input HOLD on the supervisory sequence header is set. If the input MAN on the sequence header is set, a jump is not made until the input STEP is set.

When executing the step, the elements included are executed first, after which the jump conditions are checked. If both the conditions for jump and those for step change are satisfied, the jump is made. If JPOS = 0 for the jump input concerned, a jump is made to the supervisory sequence header. If the step specified by the JPOS input does not exist, the sequence stops.

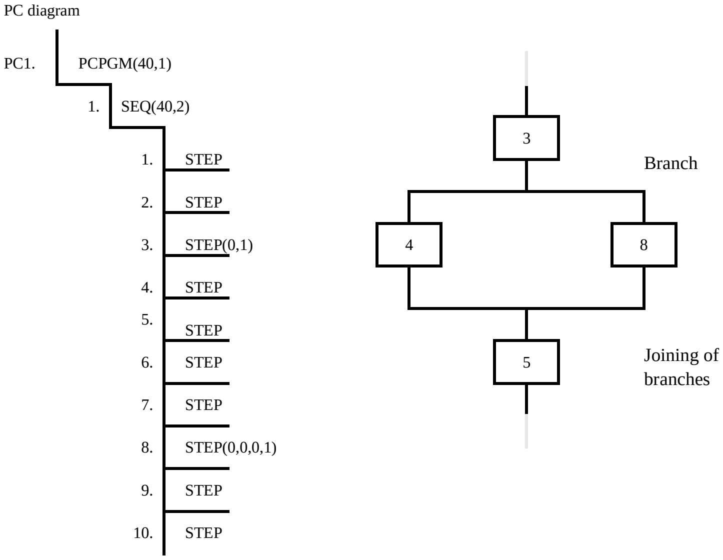
Clearing of Step

When a sequence step has been active, the data calculated remains, provided that the RUN output is not included as a condition for those data. Calculated data can be cleared by a later sequence step. One or more steps in succession can be cleared simultaneously. The step from which clearing is to be performed is specified with an integer (I) on one of the inputs RL1…RL*C3*. The step to be cleared is specified with an integer (I) on the corresponding input of RH1…RH*C3*. Clearing is performed when leaving the step with a normal step change, or with a step change, controlled by a jump condition. If RL = 0, the complete sequence is cleared, which corresponds to a reset on the sequence header. In this case RH has no effect. If the RL value does not exist, there is no reset. If RH does not exist or has a value which is smaller than RL, the reset is performed at the end of the sequence.

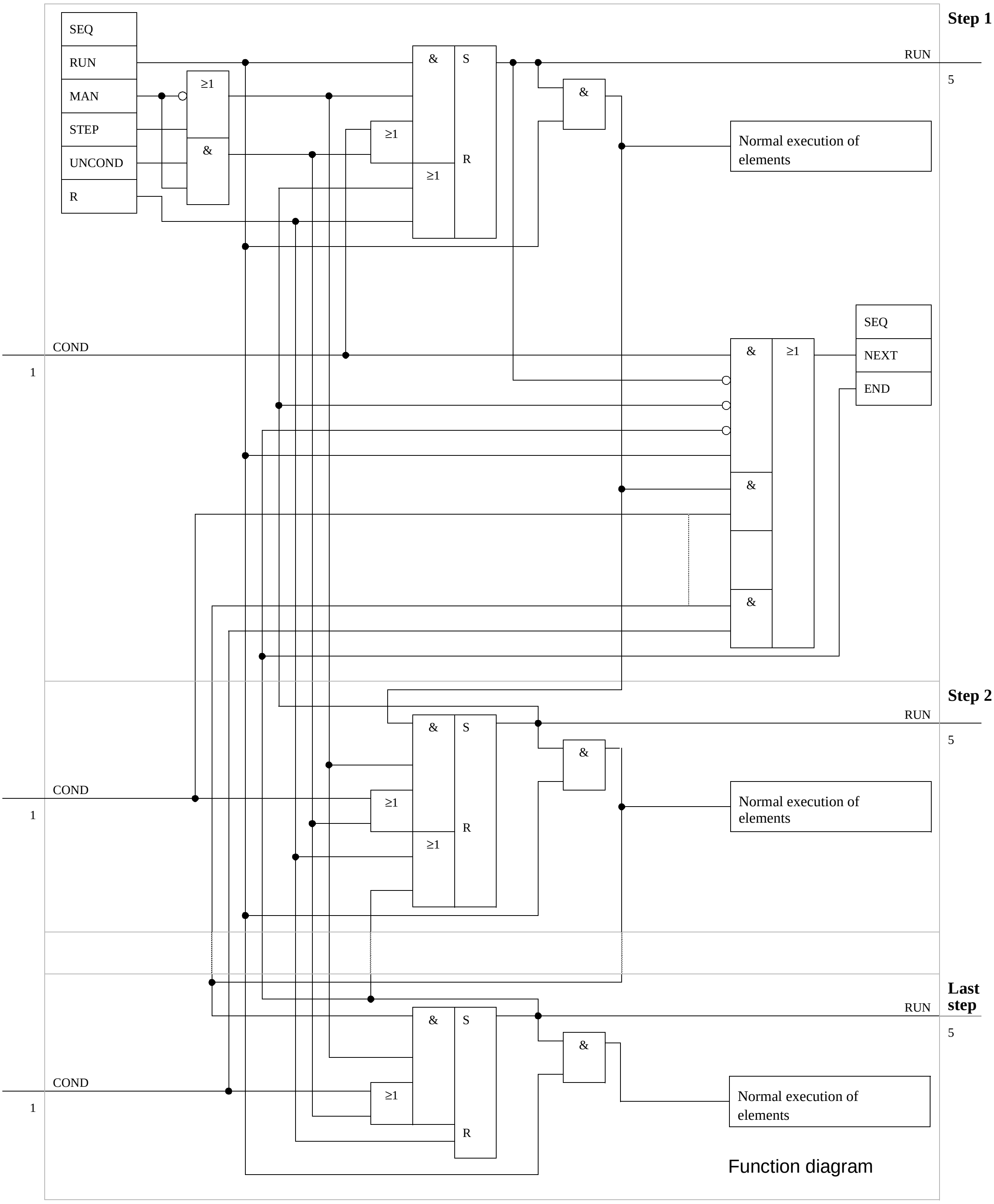
Alternative Execution Order when Joining Branched Sequences

Normally the steps are executed in the order they are build according to the PC diagram. When a sequence is branched, the jump function is used. However, when joining several sequence branches, it is better to use the alternative execution order function.

Input STEPTO then determines which step should be next step in the sequence. This step will be next step in the sequence irrespective of where it is situated in the PC diagram. Several sequence branches can have the same step determined as next step. Thus it is possible to join several sequence branches to a common step irrespective of where the steps are situated in the PC diagram. If STEPTO = 0 the execution is changed to a supervisory sequence head, and if STEPTO gives a non-existing step, the sequence is interrupted. Example: In the branched sequence in figure 116, the left branch is build as a straight, normal sequence. Branching is performed with jumps (C2 = 1 and JPOS1 = 8 in step 3) and the joining of the branches is executed with alternative execution order (C4 = 1 and STEPTO = 5 in step 8). Thus step 5 will follow step 8 when executing the sequence. The step following step 8 in the PC diagram (in this case step 9) will not affect the execution of step 8 in any way.



Example of branched sequence with joining of branches

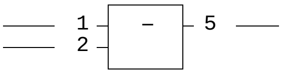


Function diagram

Subtractor SUB

Summary

SUB is used for subtraction of two integers or real numbers.



PC element SUB

Call SUB (C1)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R

Terminal Description

No	Name	Type	Description
1	–	I*C1*	Input for minuend.
2	–	I*C1*	Input for subtrahend.
20	–	O*C1*	Output for difference.

Function

The value at input 2 is subtracted from that at input 1 and the result is stored at output 20.

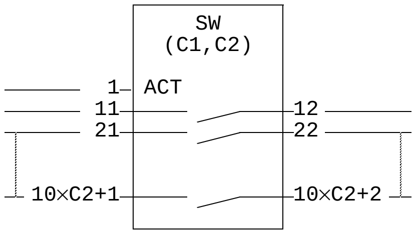
Overflow

If the maximum positive or negative values are exceeded, the output is limited to the highest or lowest representable value for the data type.

Switch SW

Summary

SW (**SW**itch) is used as a connection element for data and has an optional number of channels with closing function. The data type can be integer, real number, Boolean or time.



PC element SW

Call SW (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I , IL, R, B, T, TR
C2	Number of channels	1…9

Terminal Description

No	Name	Type	Description
1	ACT	IB	**ACT**ivate. Input for activation of the switch. When the input is set to 1 the switch is activated.
11	–	I*C1*	Input to channel 1 which is connected to the output for channel 1 when the switch is activated.
12	–	O*C1*	Output from channel 1.
21	–	I*C1*	Input to channel 2 which is connected to the output for channel 2 when the switch is activated.
22	–	O*C1*	Output from channel 2.
: 10×C2+1	: –	: I*C1*	: Input to channel C2 which is connected to the output for channel C2 when the switch is activated.
10×C2+2	–	O*C1*	Output from channel C2.

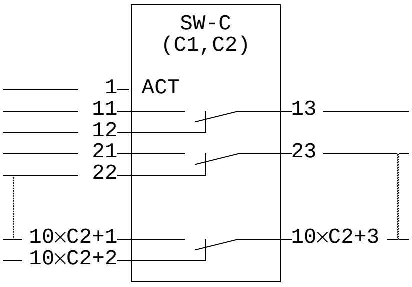
Function

When the control input ACT is 0, the output data is 0 according to the data type. When ACT is set, data comes from the inputs 11…10×C2+1.

Changeover Switch SW-C

Summary

SW-C (**SW**itch **- C**hangeover) is used as a connection element for data and has an optional number of channels with switching function. The data type can be integer, real number, Boolean or time.



PC element SW-C

Call SW-C (C1, C2)

Parameter	Significance	Permissible values
C1	Data type	I, IL, R, B, T, TR
C2	Number of channels	1…9

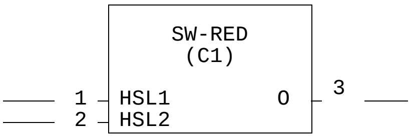
Terminal Description

No	Name	Type	Description
1	ACT	IB	**ACT**ivate. Input for activation of the switch. When the input is set to 1 the switch is activated.
11	–	I*C1*	Input for channel 1 which is connected to the output for channel 1 when the switch is activated.
12	–	I*C1*	Input to channel 1 which is connected to the output for channel 1 when the switch is not activated.
13	–	I*C1*	Output from channel 1.
21	–	I*C1*	Input to channel 2 which is connected to the output for channel 2 when the switch is not activated.
23		O*C1*	Output from channel 2.
: 10×C2+1	: –	: I*C1*	: Input to channel C2 which is connected to the output for channel C2 when the switch is activated.
10×C2+2	–	I*C1*	Input to channel C2 which is connected to the output for channel C2 when the switch is not activated.
10×C2+3	–	O*C1*	Output from channel C2.

Switch Redundancy SW-RED

Summary

The element SW-RED (**SW**itch for **RED**undancy) specifies for the planned redundancy on which High speed link (HSL) the value of the master will be transmitted.



PC element SW-RED

Call SW-RED (C1)

Parameter	Significance	Permissible values
C1	Data type of all inputs and outputs.	B, I, IL, R, T, TR

Terminal Description

No	Name	Type	Description
1	HSL1	I*C1*	Input for value from controller via **H**igh **S**peed **L**ink **1**.
2	HSL2	I*C1*	Input for value from controller via **H**igh **S**peed **L**ink **2**.
3	O	O*C1*	**O**utput.

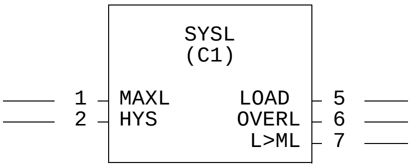
Function

If the master is connected to the High speed link 1 (HSL1), output O corresponds to input HSL1. If the master is connected to the High speed link 2 (HSL2), output O is identical to HSL2.

Element for Calculation of System Load SYSL

Summary

SYSL (SYStem Load) calculates the system load. If the system is overloaded this is indicated. Inputs and outputs for handling of maximum load may be selected with a call parameter.



PC Element SYSL

Call SYSL (C1)

Parameter	Significance	Permissible values
C1	Determines whether inputs and outputs for the maximum load is to be included in the element.	0…1

Terminal Description

No	Name	Type	Description
1(1)	MAXL	IR	**MAX**imum **L**oad. Input for maximum system load.
2(1)	HYS	IS	**HYS**teresis. Input for hysteresis for maximum system load.
5	LOAD	OR	Output for system **LOAD** in percent.
6	OVERL	OB	**OVER**Load. Output that is set to 1 when the system is fully loaded.
7(1)	L>ML	OB	**L**oad **>** **M**aximum **L**oad. Input that is set to 1 when the load exceeds the value specified at the input MAXL.

(1) These connections are only included if the call parameter C1=1.

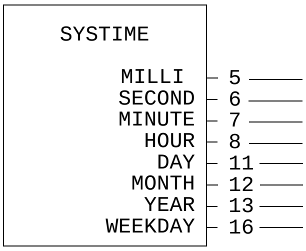
Function

If the system is fully loaded the output OVERL is set to 1. The system load in percent (%) is given at the output LOAD. If the call parameter C1 has been set to 1 the element will have inputs MAXL and HYS, for maximum load and hysteresis respectively. When the load MAXL is exceeded the output L>ML is set to one. L>ML is reset when the load lowers to below MAXL–HYS. If MAXL is higher than 100 the value 100 is used. If HYS is higher than MAXL the MAXL is used. If MAXL or HYS is negative the value 0 is used.

Time and Date Element SYSTIME

Summary

SYSTIME is used to get actual time and date



PC element SYSTIME

Call SYSTIME

Terminal Description

No	Name	Type	Description
5	MILLI	OI	Output for **MILLI**second (0…999)
6	SECOND	OI	Output for **SECOND** (0…59)
7	MINUTE	OI	Output for **MINUTE** (0…59)
8	HOUR	OI	Output for **HOUR** (0…23)
11	DAY	OI	Output for **DAY** of month (1…31)
12	MONTH	OI	Output for **MONTH** (1…12)
13	YEAR	OI	Output for **YEAR** (≥1990)
16	WEEKDAY	OI	Output for **DAY** of **WEEK** (1…7), Monday = 1, …, Sunday = 7

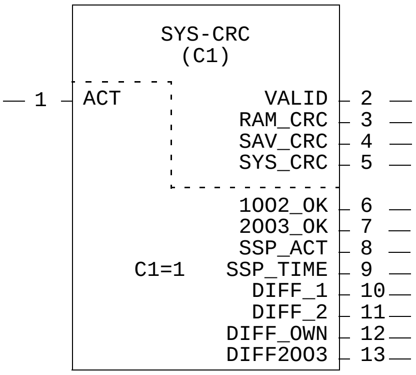
Function

Actual system time and date are output at the terminals of the PC element.

Manage Fingerprints SYS-CRC

Summary

The PC element SYS-CRC manages “fingerprints” of a multi-channel safety system. A fingerprint is a common CRC checksum taken from the operating system, the system software, all options, and the application program.



PC element SYS-CRC

Call SYS-CRC (C1)

Parameter	Significance	Permissible values
C1	Show safety part.	0…1

Terminal Description

No	Name	Type	Description
2	VALID	OB	True, if calculated CRC’s are **VALID**.
3	RAM_CRC	OI	**CRC** calculated from the application program.
4	SAV_CRC	OI	**CRC** of application program (RAM_CRC) **SAV**ed to a storage domain by service “Application Checksum → Store”.
5	SYS_CRC	OI	**CRC** of **SYS**tem software and all options.
If C1=1, the following terminals will be shown:			
1	ACT	IB	**ACT**ivates output terminal DIF2OO3.
6	1OO2_OK	OB	**1 O**ut **O**f **2** redundancy is **OK**ay. Indicates the protection system status derived from the high speed communication links.
7	2OO3_OK	OB	**2 O**ut **O**f **3** redundancy is **OK**ay. Indicates the protection system status derived from the high speed communication links.
8	SSP_ACT	OB	**S**econd **S**hutdown **P**ath **ACT**ivated.
9	SSP_TIME	OT	Indicates the remaining **S**econd **S**hutdown **P**ath **TIME** as degraded protection system before shutdown.

Terminal Description (Continued)

No	Name	Type	Description
10	DIFF_1	OB	Indicates a **DIFF**erence of the current fingerprint from protection channel **1** with its reference value stored with service “Application Checksum → Store”.
11	DIFF_2	OB	Indicates a **DIFF**erence of the current fingerprint from protection channel **2** with its reference value stored with service “Application Checksum → Store”.
12	DIFF_OWN	OB	Indicates a new fingerprint of system or application software after service “Application Checksum → Store” has been executed.
13	DIFF2OO3	OB	Indicates that the checksum of **2 O**ut **O**f **3** protection channels **DIFF**ers (only if ACT=1).

Function

The PC element detects changes in the partner Processor Module for 2-out-of-3 safety applications.

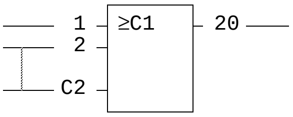
NOTE

The safety part (C1 = 1) of SYS-CRC is designed for use in redundant applications. Do not use it for non-redundant applications.

Threshold Function THRESH-L

Summary

THRESH-L (**THRESH**old **- L**ogic) is used to determine when more than, or equal to, a given number of Boolean signals are set (to 1).



PC element THRESH-L

Call THRESH-L (C1, C2)

Parameter	Significance	Permissible values
C1	Threshold value	1…C2
C2	Number of inputs	1…19

Terminal Description

No	Name	Type	Description
1	–	IB	Input for the first Boolean value.
2	–	IB	Input for the second Boolean value.
: C2	–	IB	Input for the last Boolean value.
20	–	OB	Output which is set to 1 if C1 or more of the inputs 1…C2 are set.

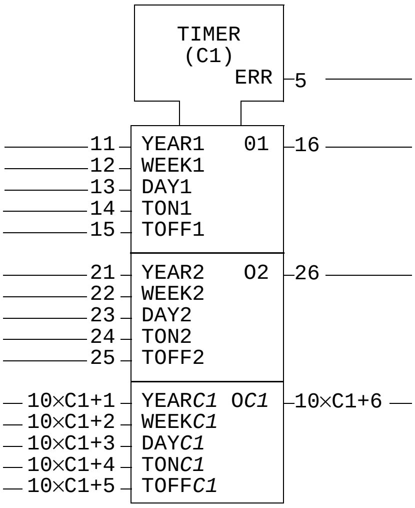
Function

The THRESH-L element is used to determine if more than, or equal to, a specified number of Boolean values in a group are set to 1. If so, the output (20) is set.

Time Selector TIMER

Summary

TIMER is used to generate a pulse with a duration determined by two times in the same (24 hour) day.



PC element TIMER

Call TIMER (C1)

Parameter	Significance	Permissible values
C1	Number of times	1…9

Terminal Description

No	Name	Type	Description
5	ERR	OB	**ERR**or. Error signal which is set to one when any of the inputs have values outside the permitted time range.
11	YEAR1	II	Input which specifies which **YEAR** the pulse is to be given from the first time element. The value 0 means a pulse is to be given each year.
12	WEEK1	II	Input which specifies which **WEEK** the pulse is to be given from the first time element. The value 0 means a pulse is to be given each week.
13	DAY1	II	Input which specifies which **DAY** of the week the pulse is to be given from the first time element. The value 0 means a pulse is to be given each day.
14	TON1	IT	**T**ime **ON 1**. Input which specifies at which time the pulse from the first time element is to begin.
15	TOFF1	IT	**T**ime **OFF 1**. Input which specifies at which time the pulse from the first time element is to end.
16	O1	OB	**O**utput **1**. Output at which a pulse appears between the times specified at the inputs TON1 and TOFF1.

Terminal Description (Continued)

No	Name	Type	Description
21	YEAR2	II	Input which specifies which **YEAR** the pulse is to be given from the second time element. The value 0 means a pulse is to be given each year.
22	WEEK2	II	Input which specifies which **WEEK** the pulse is to be given from the second time element. The value 0 means a pulse is to be given each week.
23	DAY2	II	Input which specifies which **DAY** of the week the pulse is to be given from the second time element. The value 0 means a pulse is to be given each day.
24	TON2	IT	**T**ime **ON 2**. Input which specifies at which time the pulse from the second time element is to begin.
25	TOFF2	IT	**T**ime **OFF 2**. Input which specifies at which time the pulse from the second time element is to end.
26	O2	OB	**O**utput **2**. Output at which a pulse appears between the times specified at the inputs TON2 and TOFF2.
: 10×C1+1	YEAR*C1*	II	Input which specifies which **YEAR** the pulse is to be given from time element ***C1***. The value 0 means a pulse is to be given each year.
10×C1+2	WEEK*C1*	II	Input which specifies which **WEEK** the pulse is to be given from time element ***C1***. The value 0 means a pulse is to be given each week.
10×C1+3	DAY*C1*	II	Input which specifies which **DAY** of the week the pulse is to be given from time element ***C1***. The value 0 means a pulse is to be given each day.
10×C1+4	TON*C1*	IT	**T**ime **ON *C1***. Input which specifies at which time the pulse from the last time element is to begin.
10×C1+5	TOFF*C1*	IT	**T**ime **OFF *C1***. Input which specifies at which time the pulse from the last time element is to end.
10×C1+6	O*C1*	OB	**O**utput ***C1***. Output at which a pulse appears between the times specified at the inputs TON*C1* and TOFF*C1*.

Function

TIMER is used to produce a pulse between two times within one (24 hours) day. The day is specified by year, week number and day number within the week. The duration of the pulse is determined by the inputs TON and TOFF. The generated pulse can be used to start required functions.

YEAR Input

The year during which the pulse is to be obtained is specified at the inputs YEAR1…YEAR*C1*. If the year is given as 0, a pulse is given each year. In addition to 0, years 1980 to 2099 can be given.

WEEK Input

The week during which the pulse is to be obtained is specified at the inputs WEEK1…WEEK*C1*. If the week is given as 0, a pulse is given each week. In addition to 0, week numbers 1 to 53 can be given. If week 53 is specified in a year that does not have a week 53, no pulse is produced.

DAY Input

The day of the week during which the pulse is to be obtained is specified at the inputs DAY1…DAY*C1*. If the day is given as 0, a pulse is given each day of the week. In addition to 0, day numbers 1 to 7 can be given. Day 1 refers to Monday and day 7 is Sunday.

Rules for Week and Day Numbering

Weeks and days are numbered in accordance with the ISO standard. Monday is considered to be the first day of the week. The first week which contains at least 4 days is considered to be the first week of the year. When days are referred to by year-week-day, the change of year is on week 1, day 1.

Example: 1 Jan 1995 is given as 1994 52 6

TON and TOFF Inputs

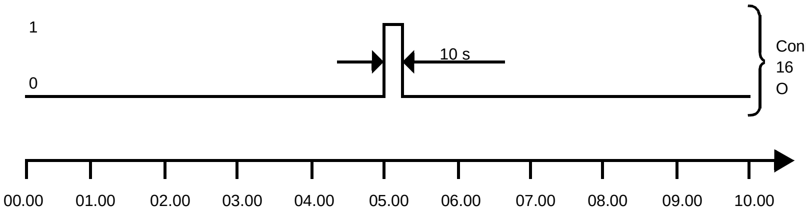
At which time a pulse is to begin or end respectively is given at the inputs TON and TOFF. TON and TOFF must be within the same 24 hour period, that is TON<TOFF. Time specifications can vary from 00:00:00.000 to 23:59:59.999. For repetition within a 24 hour period, for example each hour, external elements are required.

Examples of Selection of Day

YEAR	WEEK	DAY	Significance
0	0	0	Each day.
1980…2099	0	0	Each day during a certain year.
1980…2099	1…53	0	Each day during a certain week during a certain year.
1980…2099	0	1…7	A certain day each week during a certain year.
0	1…53	0	Each day during a certain week every year.
0	0	1…7	A certain day of each week, every year.
0	1…53	1…7	A certain day of each year.
1980…2099	1…53	1…7	A particular day.

Supervision

If any input has a value outside the permitted range, the error signal ERR is set (to 1).

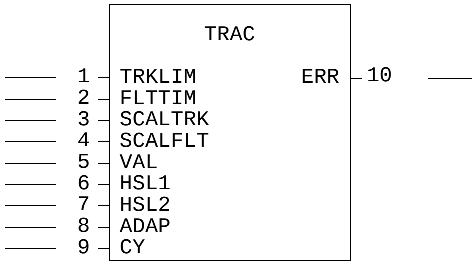


Time pulse diagram

Tracking TRAC

Summary

The element TRAC (**TRAC**king) selects for the redundancy task which one of the redundant controllers has to be tracked to the other controllers.



PC element TRAC

Call TRAC

Terminal Description

No	Name	Type	Description
1	TRKLIM	IR	**LIM**it to detect need for **TR**ac**K**ing(1).
2	FLTLIM	IR	**LIM**it to detect **F**au**LT**(1).
3	SCALTRK	IR	**SCAL**ing factor for input **TRK**LIM. Default value 1.05.
4	SCALFLT	IR	**SCAL**ing factor for input **FLT**LIM. Default value 1.2.
5	VAL	IR	**VAL**ue of the own controller(1).
6	HSL1	IR	Value from controller via HSL1 (**H**igh **S**peed **L**ink **1**)(1).
7	HSL2	IR	Value from controller via HSL2 (**H**igh **S**peed **L**ink **2**)(1).
8	ADAP	IR	**ADAP**tion variable.
9	CY	IT	The longest **CY**cle time of a tracking CONTRM. This time must not exceed 500 ms.
10	ERR	OB	**ERR**or output.

(1) This value must be entered.

Function

For redundancy, a significant variable VAL is needed which will be defined for all the 3 redundant controllers. The mean value will be calculated from the values VAL, HSL1, and HSL2. If one of the values VAL has a deviation from the mean value being greater than TRKLIM, the controller will be tracked. If the deviation is greater than FLTLIM, the controller will probably be faulty.

Because of procedures which will affect great changes of the value at VAL, asynchronous controllers will also produce values more differing from each other, the values TRKLIM and FLTLIM must be adapted to TRKLIM′ and FLTLIM′ in that case.

$$\text{TRKLIM}' = \max\,(\text{TRKLIM},\ dx/dt \times T' \times \text{SCALTRK})$$

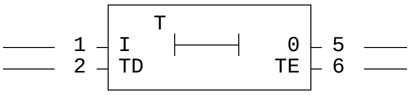
$$\text{FLTLIM}' = \max\,(\text{FLTLIM},\ dx/dt \times T' \times \text{SCALFLT})$$

where T′ is the cycle time and dx/dt is the change per time of the adaption variable ADAP or if ADAP is not applied, dx/dt is the change per time of the mean value of VAL.

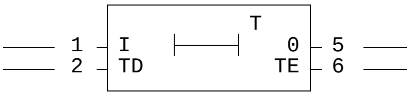
Time Delay On/Off TON, TOFF

Summary

Time delay on TON (**T**imer **ON**-delay) and time delay off TOFF (**T**imer **OFF**-delay) of Boolean variables for use in connection with combined expressions.



PC element TON



PC element TOFF

Call TON, TOFF

Terminal Description

No	Name	Type	Description
1	I	IB	**I**nput for start of time delay.
2	TD	IT	**T**ime **D**elay. Input for preset time. Max 23h 59m 59.999s.
5	O	OB	**O**utput. Output which is set (TON) or reset (TOFF) when the preset time has elapsed.
6	TE	OT	**T**ime **E**lapsed. Output which specifies how long I has been set (TON) or reset (TOFF). When the preset time (TD) has elapsed, TE stops.

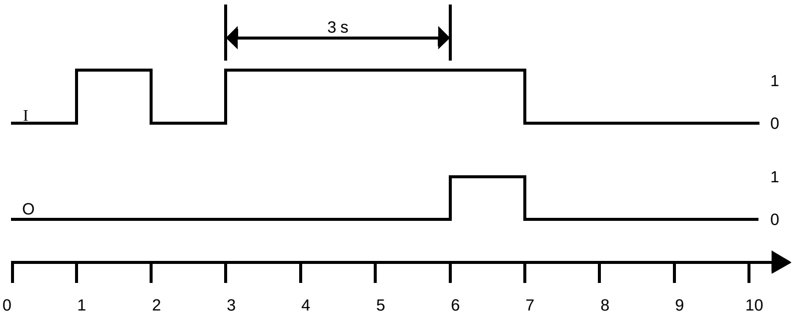
Function

TON

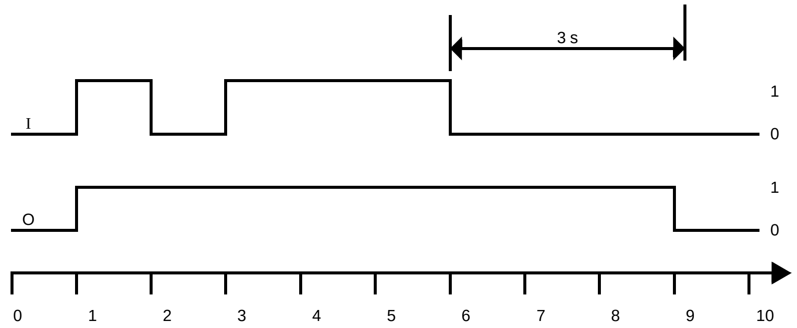
The input variable to the input I is obtained delayed at output O when the input variable changes from 0 to 1. The output signal returns immediately the input variable changes from 1 to 0.

TOFF

The input variable to the input I is obtained delayed at output O when the input variable changes from 1 to 0. The output signal is set immediately the input variable changes from 0 to 1.



Example of time diagram for TON with preset time 3 s



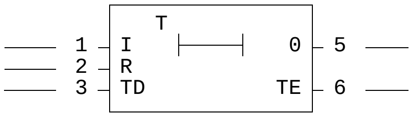
Example of time diagram for TOFF with preset time 3 s

Time Delay On with Retention

TON-RET

Summary

TON-RET (**T**ime delay **ON - RET**entive) switch-on delay with accumulating time measurement of Boolean variables for use in connection with combined expressions. TON-RET is used for example operational time monitoring.



PC element TON-RET

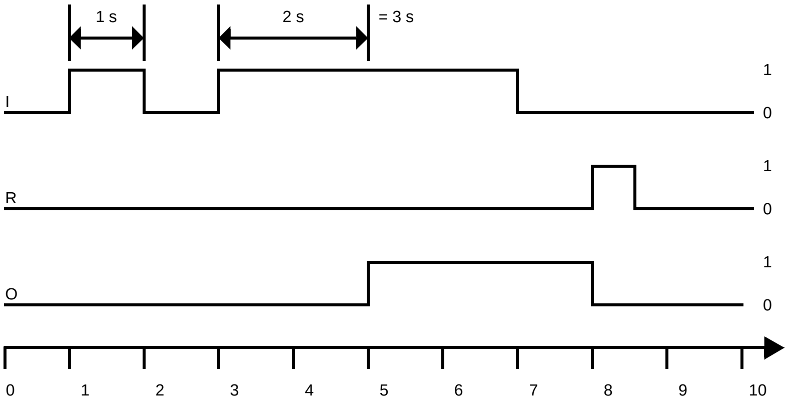
Call TON-RET

Terminal Description

No	Name	Type	Description
1	I	IB	**I**nput for start of time delay when the input changes from 0 to 1. The time circuit maintains its status even when the input returns to 0. **O**utput which is set when the preset time has elapsed.
2	R	IB	**R**eset. Input which resets the timer. R must be reset (to 0) for the timer to function.
3	TD	IT	**T**ime **D**elay. Input for preset time, that is the time during which the input I must be set (to 1). Max 23h 59m 59s.
5	O	OB	**O**utput which is set when the preset time has elapsed.
6	TE	OT	**T**ime **E**lapsed. Output which indicates the time during which I has been set to 1.

Function

The input variable to the input I is obtained delayed at the output O when the input variable changes from 0 to 1 in accordance with the time pulse diagram in figure 2. If the input variable returns to 0, the time which has elapsed remains in the timer and when the variable returns to 1, the time continues from the value which applied when the variable went to 0. The input R must be reset (to 0) when the timer functions. If R is set (to 1) both outputs O and TE are reset.

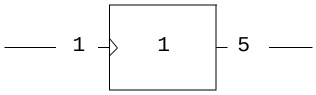


Example of time diagram with preset time 3 s

Trigger **TRIGG**

Summary

The triggering element TRIGG is used for, for example, reducing impulse times at the start of automatic procedures and for calculating functions.



PC element TRIGG

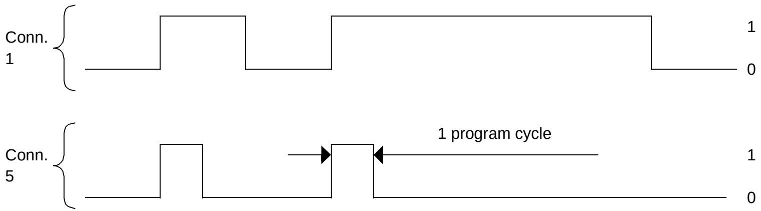
Call TRIGG

Terminal Description

No	Name	Type	Description
1	–	IB	Input for start of trigger pulse.
5	–	OB	Output for trigger pulse.

Function

When the input signal is set (to 1) the output signal is also set. The output signal is cleared during the next program cycle, irrespective of the value of the input signal. For the output signal to be set, the input signal must have been 0 during one program cycle. The output signal from the element may only be used within its own execution unit as detection of it is otherwise not certain.

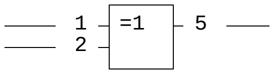


Time diagram

Exclusive Or Gate XOR

Summary

XOR (e**X**clusive **OR**) is used to generate combined expressions with Boolean variables.



PC element XOR

Call XOR

Terminal Description

No	Name	Type	Description
1	–	IB	Input.
2	–	IB	Input.
5	–	OB	Output.

Function

The output signal from the XOR element is 1 if the input signals are different and 0 if they are equal, see the table below. By inverting the output of the XOR element, a 1 is obtained if the signals are equal.

Table Input and Output Signals

1	2	5
0	0	0
0	1	1
1	0	1
1	1	0

3BDS 005 557R101

April 2000